***Título*: Content Management System**

***Volumen*: 1**
***Alumno*: Joan Casas Cervero**

***Director/Ponente*: Horacio Rodríguez Hontoria**
***Departamento*: LSI**

DATOS DEL PROYECTO

Título del proyecto: Content Management System

Nombre del estudiante: Joan Casas Cerveró
Titulación: Ingeniero Técnico en informática de Gestión
Créditos: 22,5

Director/Ponente: Horacio Rodríguez Hontoria

Departamento: LSI

MIEMBROS DEL TRIBUNAL

Presidenta: Alicia Maria Agneo Pulido

Vocal: Jordi Torres Viñals

Secretario: Horacio Rodríguez Hontoria

CALIFICACIÓN

Calificación numérica:

Calificación descriptiva:

Fecha:

Index

1 Introducción

Realizar un web puede ser un trabajo complicado y muy laborioso si no se dispone de las herramientas adecuadas. En el pasado las herramientas eran básicamente editores que permitían generar una página, que evolucionaron para incorporar el control de la estructura de la web y otras funcionalidades, pero en general estaban enfocadas más a la creación que al mantenimiento. En los últimos años se ha desarrollado el concepto de sistema de gestión de contenidos (content Management Systems o CMS). Se trata de herramientas que permiten crear y mantener un web con facilidad, encargándose de los trabajos más tediosos que hasta ahora ocupaban el tiempo de los administradores de las webs.

Teniendo en cuenta el ahorro que supone la utilización de estas herramientas, y el coste de desarrollarlas, sería lógico esperar que su precio fuera muy elevado. Eso es cierto para algunos productos comerciales, pero existen potentes herramientas de gestión de contenidos de acceso libre, disponibles con licencias de código abierto.

Los gestores de contenidos proporcionan un entorno que posibilita la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios. En cualquier entorno virtual ésta es una característica importante, que además puede ayudar a crear una comunidad cohesionada que participe más de forma conjunta.

En este artículo se describen los criterios más importantes a la hora de seleccionar un gestor de contenidos y los requerimientos en función de los objetivos que se quieran alcanzar. Por eso, se hace un breve repaso de las herramientas de código abierto que permiten construir sistemas gestores de contenido generales y se hace una particularización de aquéllas más orientadas hacia la construcción de espacios virtuales de e-learning.

1.1 Que es un CMS?

Los sistemas de gestión de contenidos (Content Management Systems o CMS) es un software que se utiliza principalmente para facilitar la gestión de webs, ya sea en Internet o en una intranet, y por eso también son conocidos como gestores de contenido web (Web Content Management o WCM). Hay que tener en cuenta, sin embargo, que la aplicación de los CMS no se limita sólo a las webs.

James Robertson (2003 b) propone una división de la funcionalidad de los sistemas de gestión de contenidos en cuatro categorías: creación de contenido, gestión de contenido, publicación y presentación.

1.1.1 Creación de contenido

Un CMS aporta herramientas para que los creadores sin conocimientos técnicos en páginas web puedan concentrarse en el contenido. Lo más habitual es proporcionar un editor de texto WYSIWYG, en el que el usuario ve el resultado final mientras escribe, al estilo de los editores comerciales, pero con un rango de formatos de texto limitado. Esta limitación tiene sentido, ya que el objetivo es que el creador pueda poner énfasis en algunos puntos, pero sin modificar mucho el estilo general del sitio web.

Hay otras herramientas como la edición de los documentos en XML, utilización de aplicaciones ofimáticas con las que se integra el CMS, importación de documentos existentes y editores que permiten añadir marcas, habitualmente HTML, para indicar el formato y estructura de un documento.

Un CMS puede incorporar una o varias de estas herramientas, pero siempre tendría que proporcionar un editor WYSIWYG por su facilidad de uso y la comodidad de acceso desde cualquier ordenador con un navegador y acceso a Internet.

Para la creación del sitio propiamente dicho, los CMS aportan herramientas para definir la estructura, el formato de las páginas, el aspecto visual, uso de patrones, y un sistema modular que permite incluir funciones no previstas originalmente.

1.1.2 Gestión de contenido

Los documentos creados se depositan en una base de datos central donde también se guardan el resto de datos de la web, como son los datos relativos a los documentos (versiones hechas, autor, fecha de publicación y caducidad, etc.), datos y preferencias de los usuarios, la estructura de la web, etc.

La estructura de la web se puede configurar con una herramienta que, habitualmente, presenta una visión jerárquica del sitio y permite modificaciones. Mediante esta estructura se puede asignar un grupo a cada área, con responsables,

editores, autores y usuarios con diferentes permisos. Eso es imprescindible para facilitar el ciclo de trabajo (workflow) con un circuito de edición que va desde el autor hasta el responsable final de la publicación. El CMS permite la comunicación entre los miembros del grupo y hace un seguimiento del estado de cada paso del ciclo de trabajo.

1.1.3 Publicación

Una página aprobada se publica automáticamente cuando llega la fecha de publicación, y cuando caduca se archiva para futuras referencias. En su publicación se aplica el patrón definido para toda la web o para la sección concreta donde está situada, de forma que el resultado final es un sitio web con un aspecto consistente en todas sus páginas. Esta separación entre contenido y forma permite que se pueda modificar el aspecto visual de un sitio web sin afectar a los documentos ya creados y libera a los autores de preocuparse por el diseño final de sus páginas.

1.1.4 Presentación

Un CMS puede gestionar automáticamente la accesibilidad del web, con soporte de normas internacionales de accesibilidad como WAI, y adaptarse a las preferencias o necesidades de cada usuario. También puede proporcionar compatibilidad con los diferentes navegadores disponibles en todas las plataformas (Windows, Linux, Mac, Palm, etc.) y su capacidad de internacionalización lo permite adaptarse al idioma, sistema de medidas y cultura del visitante.

El sistema se encarga de gestionar muchos otros aspectos como son los menús de navegación o la jerarquía de la página actual dentro del web, añadiendo enlaces de forma automática. También gestiona todos los módulos, internos o externos, que incorpore al sistema. Así por ejemplo, con un módulo de noticias se presentarían las novedades aparecidas en otro web, con un módulo de publicidad se mostraría un anuncio o mensaje animado, y con un módulo de foro se podría mostrar, en la página principal, el título de los últimos mensajes recibidos. Todo eso con los enlaces correspondientes y, evidentemente, siguiendo el patrón que los diseñadores hayan creado.

1.2 Necesidad de un CMS

En el apartado anterior se han presentado bastantes motivos para ver la utilidad de un sistema que gestione un entorno web, pero se podría pensar que no es necesario para un web relativamente pequeña o cuando no se necesitan tantas funcionalidades. Eso sólo podría ser cierto para una web con unas pocas páginas estáticas para el que no se prevea un crecimiento futuro ni muchas actualizaciones, lo que no es muy realista. En cualquier otro caso, la flexibilidad y escalabilidad que permiten estos sistemas, justifican su utilización en prácticamente cualquier tipo de web.

Muchos usuarios particulares utilizan CMS gratuitos para elaborar y gestionar sus webs personales, obteniendo webs dinámicas llenos de funcionalidades. El resultado que obtienen es superior al de algunas empresas que se limitan a tener páginas estáticas que no aportan ningún valor añadido.

Éstos son algunos de los puntos más importantes que hacen útil y necesaria la utilización de un CMS:

Inclusión de nuevas funcionalidades en la web. Esta operación puede implicar la revisión de multitud de páginas y la generación del código que aporta las funcionalidades. Con un CMS eso puede ser tan simple como incluir un módulo realizado por terceros, sin que eso suponga muchos cambios en la web. El sistema puede crecer y adaptarse a las necesidades futuras.

Mantenimiento de gran cantidad de páginas. En una web con muchas páginas hace falta un sistema para distribuir los trabajos de creación, edición y mantenimiento con permisos de acceso a las diferentes áreas. También se tienen que gestionar los metadatos de cada documento, las versiones, la publicación y caducidad de páginas y los enlaces rotos, entre otros aspectos.

Reutilización de objetos o componentes. Un CMS permite la recuperación y reutilización de páginas, documentos, y en general de cualquier objeto publicado o almacenado.

Páginas interactivas. Las páginas estáticas llegan al usuario exactamente como están almacenadas en el servidor web. En cambio, las páginas dinámicas no existen en el servidor tal como se reciben en los navegadores, sino que se generan según las peticiones de los usuarios. De esta manera cuando por ejemplo se utiliza un buscador, el sistema genera una página con los resultados que no existían antes de

la petición. Para conseguir esta interacción, los CMS conectan con una base de datos que hace de repositorio central de todos los datos de la web.

Cambios del aspecto de la web. Si no hay una buena separación entre contenido y presentación, un cambio de diseño puede comportar la revisión de muchas páginas para su adaptación. Los CMS facilitan los cambios con la utilización, por ejemplo, del estándar CSS (Cascading Style Sheets u hojas de estilo en cascada) con lo que se consigue la independencia de presentación y contenido.

Consistencia de la web. La consistencia en un web no quiere decir que todas las páginas sean iguales, sino que hay un orden (visual) en vez de caos. Un usuario nota enseguida cuándo una página no es igual que el resto de las de la misma web por su aspecto, la disposición de los objetos o por los cambios en la forma de navegar. Estas diferencias provocan sensación de desorden y dan a entender que la web no lo han diseñado profesionales. Los CMS pueden aplicar un mismo estilo en todas las páginas con el mencionado CSS, y aplicar una misma estructura mediante patrones de páginas.

Control de acceso. Controlar el acceso a una web no consiste simplemente en permitir la entrada a la web, sino que comporta gestionar los diferentes permisos a cada área de la web aplicados a grupos o individuos.

1.3 CMS Comerciales y de codigo abierto

Se puede hacer una primera división de los CMS según el tipo de licencia escogido. Por una parte están los CMS comercializados por empresas que consideran el código fuente un activo más que tienen que mantener en propiedad, y que no permiten que terceros tengan acceso. Por la otra tenemos los de código fuente abierto, desarrollados por individuos, grupos o empresas que permiten el acceso libre y la modificación del código fuente.

La disponibilidad del código fuente posibilita que se hagan personalizaciones del producto, correcciones de errores y desarrollo de nuevas funciones. Este hecho es una garantía de que el producto podrá evolucionar incluso después de la desaparición del grupo o empresa creadora.

Algunas empresas también dan acceso al código, pero sólo con la adquisición de una licencia especial o después de su desaparición. Generalmente las

modificaciones sólo pueden hacerlas los mismos desarrolladores, y siempre según sus prioridades.

Los CMS de código abierto son mucho más flexibles en este sentido, pero se podría considerar que la herramienta comercial será más estable y coherente al estar desarrollada por un mismo grupo. En la práctica esta ventaja no es tan grande, ya que los CMS de código abierto también están coordinados por un único grupo o por empresas, de forma similar a los comerciales.

Utilizar una herramienta de gestión de contenidos de código abierto tiene otra ventaja que hace decidirse a la mayoría de usuarios: su coste. Habitualmente todo el software de código abierto es de acceso libre, es decir, sin ningún coste en licencias. Sólo en casos aislados se hacen distinciones entre empresas y entidades sin ánimo de lucro o particulares. En comparación, los productos comerciales pueden llegar a tener un coste que sólo una gran empresa puede asumir.

En cuanto al soporte, los CMS comerciales acostumbran a dar soporte profesional, con un coste elevado en muchos casos, mientras que los de código abierto se basan más en las comunidades de usuarios que comparten información y solución a los problemas. Las formas de soporte se pueden mezclar, y así encontramos CMS de código abierto con empresas que ofrecen servicios de valor añadido y con activas comunidades de usuarios. En el caso comercial también sucede, pero el coste de las licencias hace que el gran público se decante por otras opciones y por lo tanto las comunidades de soporte son más pequeñas.

Un problema que acostumbra a tener el software de código abierto es la documentación, generalmente escasa, dirigida a usuarios técnicos o mal redactada. Este problema se agrava en el caso de los módulos desarrollados por terceros, que no siempre incorporan las instrucciones de su funcionamiento de forma completa y entendible.

En el mercado hay CMS de calidad tanto comerciales como de código abierto. Muchos CMS de código abierto están poco elaborados (aunque en plena evolución), pero también lo encontramos entre los comerciales. En definitiva, un buen CMS de código abierto es mucho más económico que su homólogo comercial, con la ventaja de disponer de todo el código fuente y de una extensa comunidad de usuarios.

Por todos estos motivos, y como apuesta por la filosofía del software libre, en este trabajo sólo se presentan algunos CMS de código abierto.

1.4 Historia de los CMS

A principios de los años noventa, el concepto de sistemas de gestión de contenidos era desconocido. Algunas de sus funciones se realizaban con aplicaciones independientes: editores de texto y de imágenes, bases de datos y programación a medida.

Ya en el año 1994 Illustra Information Technology utilizaba una base de datos de objetos como repositorio de los contenidos de una web, con el objetivo de poder reutilizar los objetos y ofrecía a los autores un entorno para la creación basado en patrones. La idea no cuajó entre el público y la parte de la empresa enfocada a la Web fue comprada por AOL, mientras que Informix adquirió la parte de bases de datos.

RedDot es una de las empresas pioneras que empezó el desarrollo de un gestor de contenidos el año 1994. No fue hasta a finales del año siguiente que presentaron su CMS basado en una base de datos.

Entre los CMS de código abierto uno de los primeros fue Typo 3, que empezó su desarrollo el año 1997, en palabras de su autor, Kasper Skårhøj, "antes de que el término gestión de contenidos fuera conocido sobradamente".

PHPNuke, la herramienta que popularizó el uso de estos sistemas para las comunidades de usuarios en Internet, se empezó a desarrollar en el año 2000. La primera versión supuso tres semanas de trabajo al creador, rescribiendo el código de otra herramienta, Thatware.

1.4.1 Presente y futuro de los CMS

En la actualidad, aparte de la ampliación de las funcionalidades de los CMS, uno de los campos más interesantes es la incorporación de estándares que mejoran la compatibilidad de componentes, facilitan el aprendizaje al cambiar de sistema y aportan calidad y estabilidad.

Algunos de estos estándares son CSS, que permite la creación de hojas de estilo; XML, un lenguaje de marcas que permite estructurar un documento; XHTML, que es un subconjunto del anterior orientado a la presentación de documentos vía web;

WAI, que asegura la accesibilidad del sistema; y RSS, para sindicar contenidos de tipo noticia.

También las aplicaciones que rodean los CMS acostumbran a ser estándar, como los servidores web Apache y ISS; los lenguajes PHP, Perl y Python; y las bases de datos MySQL y PostgreSQL. La disponibilidad para los principales sistemas operativos de estas aplicaciones y módulos, permite que los CMS puedan funcionar en diversas plataformas sin muchas modificaciones.

Sobre el futuro de los CMS, Robertson (2003a) apunta que:

Los CMS se convertirán en un artículo de consumo, cuando los productos se hayan establecido y más soluciones lleguen al mercado. Eso provocará una disminución de los precios en los productos comerciales y una mayor consistencia en las funcionalidades que ofrecen.

En este entorno, muchas empresas que implementan webs tendrán que cerrar.

Muchos proyectos fracasarán por no ajustarse a los estándares y no entender conceptos como usabilidad, arquitectura de la información, gestión del conocimiento y contenido.

El campo de los gestores de contenido madurará hasta conseguir un alto grado de consistencia y profesionalismo.

Se adoptarán estándares en el almacenaje, estructuración y gestión del contenido.

Se producirá una fusión entre gestión de contenidos, gestión de documentos y gestión de registros.

También se puede añadir la incorporación de sistemas de e-learning y gestión del conocimiento, y en los entornos de intranet corporativa, la posibilidad de acceder a otras fuentes de datos como por ejemplo sistemas de soporte de decisiones (Decision Support Systems o DSS). El campo de los CMS de código abierto tendría que seguir un desarrollo similar.

1.5 Los CMS en el e-learning

El e-learning tiene unas necesidades específicas que un CMS general no siempre cubre, o si lo hace, no da las mismas facilidades que una herramienta creada específicamente por esta función.

En general, los sistemas de gestión del aprendizaje (Learning Management Systems o LMS) facilitan la interacción entre los profesores y los estudiantes, aportan herramientas para la gestión de contenidos académicos y permiten el seguimiento y la valoración de los estudiantes. Es decir, facilitan una translación del modelo real en el mundo virtual.

Un buen ejemplo de sistema de gestión de cursos es Moodle , uno de los más conocidos con licencia de código abierto. Sus características pueden servir para concretar algunas de las funcionalidades que se esperan de este tipo de herramientas:

Administración de profesores y alumnos.

Aulas virtuales que contienen toda la información de un curso y permiten la comunicación con foros o con chats.

Creación, mantenimiento y publicación del material de un curso, con soporte de diferentes formatos, incluidos audio y vídeo.

Talleres virtuales.

Exámenes y tests con valoraciones.

Trabajos con fecha de límite de entrega y aviso al profesor en caso de incumplimiento.

Seguimiento estadístico de las acciones del estudiante.

Estos sistemas son diferentes a los CMS, tanto por el objetivo como por las características, pero actualmente empiezan a incluir capacidades de los sistemas de gestión de contenidos. Con la integración de las dos herramientas nace un nuevo concepto, los LCMS (Learning Content Management Systems o sistemas de gestión de contenidos para el aprendizaje).

1.6 Criterios de selección

Antes de empezar el proceso de selección de un CMS concreto, hay que tener claros los objetivos de la web, teniendo en cuenta al público destinatario, y estableciendo una serie de requerimientos que tendría que poder satisfacer el CMS.

La siguiente lista está basada en las funciones principales de los CMS expuestas anteriormente, las indicaciones de Robertson, J. (2002) y una recopilación de los requerimientos básicos de una web.

Código abierto. Por los motivos mencionados anteriormente, el CMS tendría que ser de código fuente abierto (o libre).

Arquitectura técnica. Tiene que ser fiable y permitir la escalabilidad del sistema para adecuarse a futuras necesidades con módulos. También tiene que haber una separación de los conceptos de contenido, presentación y estructura que permita la modificación de uno de ellos sin afectar a los otros. Es recomendable, pues, que se utilicen hojas de estilo (CSS) y patrones de páginas.

Grado de desarrollo. Madurez de la aplicación y disponibilidad de módulos que le añaden funcionalidades.

Soporte. La herramienta tiene que tener soporte tanto por parte de los creadores como por otros desarrolladores. De esta manera se puede asegurar de que en el futuro habrá mejoras de la herramienta y que se podrá encontrar respuesta a los posibles problemas.

Posición en el mercado y opiniones. Una herramienta poco conocida puede ser muy buena, pero hay que asegurar de que tiene un cierto futuro. También son importantes las opiniones de los usuarios y de los expertos.

Usabilidad. La herramienta tiene que ser fácil de utilizar y aprender. Los usuarios no siempre serán técnicos, por lo tanto hace falta asegurar que podrán utilizar la herramienta sin muchos esfuerzos y sacarle el máximo rendimiento.

Accesibilidad. Para asegurar la accesibilidad de una web, el CMS tendría que cumplir un estándar de accesibilidad. El más extendido es WAI (Web Accessibility Initiative) del World Wide Web Consortium.

Velocidad de descarga. Teniendo en cuenta que no todos los usuarios disponen de líneas de alta velocidad, las páginas se tendrían que cargar rápidamente o dar la opción de cargar una página alternativa con HTML básico.

Funcionalidades. No se espera que todas las herramientas ofrezcan todas las funcionalidades, ni que éstas sean las únicas que tendrá finalmente la web. Entre otras:

Editor de texto WYSIWYG a través del navegador.

Herramienta de búsqueda.

Comunicación entre los usuarios (foros, correo electrónico, chat).

Noticias.

Artículos.

Ciclo de trabajo (workflow) con diferentes perfiles de usuarios y grupos de trabajo.

Fechas de publicación y caducidad.

Webs personales.

Carga y descarga de documentos y material multimedia.

Avisos de actualización de páginas o mensajes en los foros, y envío automático de avisos por correo electrónico.

Envío de páginas por correo electrónico.

Páginas en versión imprimible.

Personalización según el usuario.

Disponibilidad o posibilidad de traducción al catalán y al castellano.

Soporte de múltiples formados (HTML, Word, Excel, Acrobat, etc.).

Soporte de múltiples navegadores (Internet Explorer, Netscape, etc.).

Soporte de sindicación (RSS, NewsML, etc.).

Estadísticas de uso e informes.

Control de páginas caducadas y enlaces rotos.

1.7 Motivación

La elección de mi proyecto fin de carrera no fue una tarea sencilla. Primeramente me dirigí a la bolsa de proyectos ofertados por la FIB. Aunque existe una gran cantidad de proyectos en esta bolsa, no encontré ninguno que llamase mi atención. El motivo no era la temática de las propuestas si no que la presentación y explicación de éstos era bastante escasa.

Una vez consultada esta bolsa pensé en buscar un PFC mediante empresa y así contar con la posibilidad de estar remunerado. Después de un par de contactos fallidos decidí pensar por mi cuenta un tema en el realmente yo tuviera un interés.

En este momento se me presentaba la difícil tarea de pensar un tema motivador y que a la vez cumpliese los objetivos formativos de un PFC. Esto me llevo varios días, hasta que llegué a la idea de realizar mi propio CMS (Content Management System). Hasta la fecha, he utilizado en varias ocasiones este tipo de herramientas para confeccionar y desarrollar entornos web. Al contactar con estas herramientas se descubren ciertos aspectos a mejorar, con lo que encontré interesante realizar una aplicación donde mejorar los detalles que había echado de menos en otras aplicaciones. Además, si en el futuro realizase una web por cuenta propia, podría incluir en el precio esta aplicación CMS propia.

Tomada ya la decisión de la temática de mi trabajo, el siguiente paso era decidir cómo desarrollarlo. Pensando en varias tecnologías, me di cuenta de que la mejor opción era decantarse por el software libre. De esta forma el estudio económico del proyecto no se encarecería y el cliente final no se vería obligado a pagar ninguna licencia para poder usar la herramienta.

La elección del tutor de mi proyecto fue encaminada en función de las asignaturas cursadas durante la carrera. El departamento de LSI ofrece una asignatura, PROP, en la que se realiza un "proyecto" propiamente dicho. La experiencia con el tutor de esta asignatura fue buena con lo que no dudé en ponerme en contacto con él y exponerle mi situación. Él muy amablemente, apoyó mi idea y a partir de aquí empezó la realización de este proyecto.

2 Informe de definición

2.1 Razón y oportunidad del proyecto

El mundo de la informática y en concreto el mundo web es un escenario en completa y constante evolución y es obvio que las tecnologías o herramientas que un día fueron punteras si no se evolucionan se quedan desfasadas y dejan de usarse, por eso es importantísimo que cualquier herramienta esté siempre en constante evolución y reinventándose a sí misma con la finalidad de optimizarla o añadir seguridad o incluso nuevas funcionalidades o cualquier otro servicio que el mercado al que vaya dirigido necesite. Este hecho obliga a los creadores y colaboradores a estar siempre alerta de los últimos cambios y avances de la tecnología en la que trabajan.

En este avance continuo todos juegan un papel importante ya sean los usuarios reportando fallos, administradores que sugieren cosas a mejorar o los colaboradores que siempre tienen algo en mente para implementar y así optimizar en todo lo que se pueda, todo esto sucede en todas aquellas herramientas que sean software libre, es por eso que la gran mayoría no lo son si mas no los CMS más usados son siempre de software libre ya que las empresas que ofertan herramientas de este tipo con ánimo de lucro no creo que puedan llegar a estos niveles de desarrollo ni investigación.

Es por esta situación de mercado por la que he visto la oportunidad de desarrollar una nueva herramienta ofreciendo funcionalidades que yo no he encontrado en otras herramientas que he usado del mismo sector.

2.2 Situación actual

La situación actual en lo que al tema de CMS se refiere está muy reñido debido a la gran oferta que hay a continuación. Adjunto una imagen donde podéis observar algunos de los CMS más usados en todo el mundo marcando las funcionalidades de las que dispone cada uno de ellos.

	Drupal	EZ-Publish	Geeklog	Mambo	Midgard	Moodle	OpenCMS	PHPNuke	Plone	PostNuke	Slash	Tiki	Typo 3	WebGUI	Xoops
Versión	4.3.2	3.3	1.3.9	4.5		1.1.1	5.0.1	7.2	2.0	0.726	2.2.6	1.8.1	3.6.0	6.0	2.0.6
Requerimientos															
Servidor Web	Apache		Apache / IIS MySQL / otros	Apache	Apache		Tomcat Servlet MySQL / otros Java JSP / PHPML	Apache / IIS MySQL / otros	Apache / IIS / Zope	Apache / IIS	Apache	Apache / IIS MySQL / otros	Apache / IIS	Apache / IIS	Apache / Otros
Base de datos	MySQL		PHP Unix	MySQL	MySQL	MySQL / postgresql	Unix / Win	PHP Unix / Win	Zope	MySQL	MySQL	PHP Unix / Win	PHP Unix / Win	MySQL	MySQL
Lenguaje	PHP		PHP	PHP	PHP	Perl			Python	PHP	Perl	PHP	PHP	Perl	PHP
SO	Unix	Unix	Unix	Unix / Win	Unix				Unix / Win	Unix / Win	Unix	Unix / Win	Unix / Win	Unix / Win	
Soporte															
Ayuda contextual							Si	No	Si	Si		Limitado	Si	Si	
CMF / API		Si			Si		Si	Si	Si	Si		No	Si	Si	
Foro y/o lista de correo							Si	Si	Si	Si		Si	Si	Si	
Trabajo en grupo															
Aprobación contenido				Si	Si		Si	No	Si	Si		Limitado	Si	No	
Control sesión							No	No	No	No		No	Si	Si	
Permisos por recurso							Si	Limitado	Si	Si		Si	Si	Si	
Versiones				Si	Si		Si	No	Si	No		Si	Limitado	Si	
Ciclo de trabajo (Workflow)	Si			Si			Si	No	Si	No	Perl	Si	Limitado	No	
Seguimiento proyectos							No	No	Si	Si		No	No	Si	
Usuarios/autores															
Deshacer									Si	No		Limitado	No	Si	
Editor WYSIWYG				Si	Si		IE	No	Si	Si		Si	Si	Si	
Ficheros upl/dowload							No	Si	Si	Si		Si	Si	Si	Si
Página personalizada							No	No	Limitado	Si		Si	Si	No	
Características															
Accesabilidad WAI									W3C AA	No		Limitado	Si	Si	
Área de test							Si	No	Si	No		Si	Si	Si	
Auditoria							Si	No	Si	No		Limitado	Si	Si	
Backup base datos				Si								Si			
Cache	Si						Si	Si				Si			
Búsqueda		Si	Si	Si	Si		Si	Si							
Contenido programado				Si	Si		Si	No	Si	Si		Si	Si	Si	
Correo (email) a foro							No	No	Si	No		Si	Si	No	
Estadísticas				Si			No	Si	No	Si		Si	Si	Limitado	
Gestión centralizada de ficheros							Si	No	Si	Si		Si	Si	Si	
Gestión publicidad				Si			No	Si	No	Si		Si	Si	No	
Gestión patrones							Si	Limitado	Si	No		Limitado	Si	Si	
Grupos de usuarios					Si										Si
Informes de bases de datos							No	No	No	Si		No	Si	Si	
Internacionalización			Si			Si	Si	Si	Si	Si		Si	Si	Si	
Español			Si			Si									
Links entrantes							No	No	No	Si		No	Si	No	
Lenguaje de macros							No	No	Limitado	No		Limitado	Si	Si	
Lenguaje de patrones							No	No	Si	No		Si	Si	Si	
Metadatos	Si			Si		Si									
Módulos externos	Si	Si	Si	Si			Si	No	Si	Si		Si	Si	Si	Si
Niveles de interficie según usuario							No	No	No	No		RSS	RSS	RSS	
Sindicación	RSS		RSS	RSS	RSS				RSS	RSS					

.

2.3 Objetivos

El objetivo principal de este proyecto es el de desarrollar e implementar una herramienta que permita y facilite a los desarrolladores y administradores de portales webs su uso y actualización de la misma. Todos los objetivos los podemos dividir en dos grandes bloques: Objetivos Primarios y Objetivos Secundarios

Mas concretamente los objetivos son:

Objetivos Principales (propios del pfc)

- Mantenimiento de entidades (crear, modificar y eliminar)
- Mantenimiento de contenido en funcion de cada entidad (creación, modificación y eliminación)
- Publicar o despublicar contenido y entidades
- Agrupar contenido por grupo y subgrupo
- Agrupar entidades por categorias
- Relacionar contenido n <-> n
- Crear plantillas de resumen
- Crear plantillas estaticas
- Crear plantillas de detalle
- Mantenimiento de usuarios de la aplicación
- Buscador de contenido y documentos

Objetivos Secundarios (obejtivos a realizar una vez finalizado el pfc)

- Multidioma
- Seguridad por documento, por grupo y subgrupo, por usuario y por plantilla
- Agregar mas roles de usuarios incluso el usuario web
- Workflow
- Estadisticas por documento y por plantilla

2.4 Beneficios esperados

Los beneficios que espero de este proyecto son:

- Agilizar la creación de paginas web pese a que sean complejas o dispongan de mucha información
- Agilizar la edición y mantenimiento de contenido en la web
- Hacer fácil la acción de rediseñar la web sin perder la estructura de la misma
- Implicar al cliente en la edición del contenido de la web lo que le hará sentirse parte del proyecto con el fin de motivarle
- Mantener la información organizada
- La opción de poder despublicar informacón que en ese momento no queremos mostrar
- Poder filtrar información

2.6 Análisis de impacto

A. **En los clientes:** Los posibles clientes finales de esta aplicación la utilizarán como medio para tener actualizada web en todo momento y de esta manera poder ofrecer un servicio correcto y adecuado en cada momento, de este modo el cliente ve la posibilidad de que su web no se quede muerta una vez realizada y se convierte así en una web dinámica en la que podrá añadir y administrar contenido a su antojo.

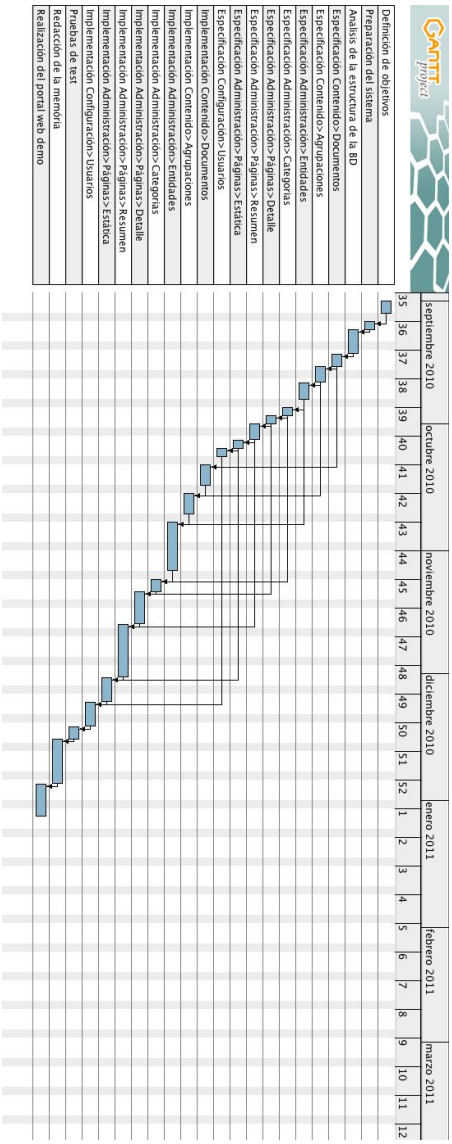
B. **En los desarrolladores web:** Los diseñadores y desarrolladores web que utilicen esta aplicación verán como el tiempo empleado para desarrollar una web se ve mermado notablemente puesto que este sistema de trabajo agiliza el trabajo monótono con el hecho de la utilización de las plantillas para diferentes tipos de documentos o incluso de entidades. Con esta aplicación será mucho mas fácil el poder hacer un modificación en cualquier web desarrollada con esta tecnología.

3 Panificación y estudio económico

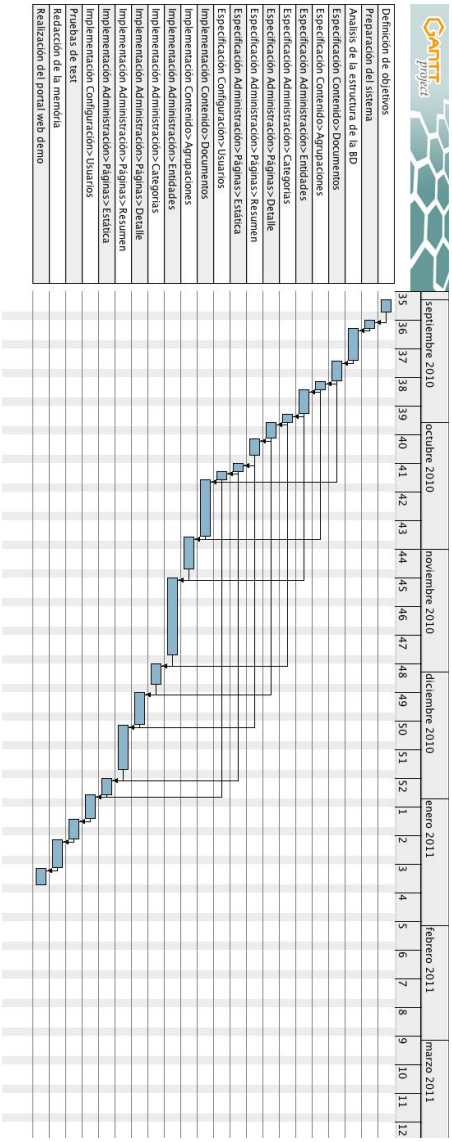
3.1 Planificación

En este apartado se presentaran las planificaciones que en un primer momento se hicieron para confeccionar un calendario aproximado de lo que se esperaba del proyecto. A continuación se mostrará la planificación resultante viéndose las diferencias diferencias entre la etapa de inicio y la de fin, todo esto reflejado en dos diagramas de gantt que muestro a continuación

3.1.1 Digrama de gantt inicial



3.1.2 Diagrama de gantt final



3.2 Estudio económico

En este apartado se realiza un análisis económico, haciendo unos cálculos y unas estimaciones para determinar que coste tendría realizar el proyecto en un ámbito fuera del marco educativo.

3.2.1 Coste del hardware

El coste de los recursos de hardware utilizados se basan en la utilización de un ordenador como el que he usado, que pondré como ejemplo, durante la realización del proyecto, y un servidor que hospedará la aplicación que se ha desarrollado junto a la web de demostración. Además del hardware, el hospedaje de la aplicación y de la web de demostración, hacer que la aplicación sea accesible desde la red también tiene un coste extra para la contratación de una conexión a internet y un dominio para ofrecer acceso a la misma.

Producto	Precio
Ordenador de sobremesa: Intel core i5 2.66Ghz 4Gb de ram 500GB HDD GT 330M	745,00€
Hosting	50€
Dominio	10€
Conexión a internet	40€
Total	845,00€

3.2.2 Coste del software

Para la realización de este proyecto se ha intentado que todas las herramientas que he usado sean de software libre con la finalidad de no encarecer el coste total estimado de la aplicación.

- Sistema operativo Fedora core

- Netbeans i kdevelop como entornos de programación

- Gimp para la edición de imágenes

3.2.3 Coste de los recursos humanos

En el proyecto realizado, además de dedicarle una media jornada de dedicación al proyecto final de carrera propiamente dicho, también he dedicado la otra mitad de la jornada como becario para desarrollar la aplicación.

Si se tiene en cuenta el precio/hora que se marca a un becario es alrededor de 6,25€, y se contempla el número total de horas que se han realizado durante el proyecto entre la media jornada del PFC y de la beca, éstas suman un total de 1104 horas dedicas al proyecto, con lo que se obtiene un coste total para los recursos humanos de 6900€.

3.2.4 Coste total

Haciendo un cálculo con los costes presentados anteriormente, el coste global del proyecto seria el siguiente:

Producto	Precio
Hardware	845,00€
Software	0€
Recursos humanos	6.900€
Total	7.745,00€

4 Plataforma tecnológica

Para llevar cabo este proyecto se han tenido que usar una serie de tecnologías, la mayoría de ellas enfocadas al desarrollo especifico de la web, pero también otras de estas tecnologías a un propósito mas general

Durante la descripción de éstas haré un especial énfasis en algunos aspectos de las tecnologías que han sido importantes o si mas no interesantes para el desarrollo de

esta aplicación, como también describiré los posibles aspectos negativos que me ha reportado alguna de estas tecnologías.

4.1 HTML

4.1.1 Historia

La historia completa de HTML es tan interesante como larga, por lo que a continuación se muestra su historia resumida a partir de la información que se puede encontrar en la Wikipedia.

El origen de HTML se remonta a 1980, cuando el físico Tim Berners-Lee, trabajador del CERN (*Organización Europea para la Investigación Nuclear*) propuso un nuevo sistema de *"hipertexto"* para compartir documentos.

Los sistemas de *"hipertexto"* habían sido desarrollados años antes. En el ámbito de la informática, el *"hipertexto"* permitía que los usuarios accedieran a la información relacionada con los documentos electrónicos que estaban visualizando. De cierta manera, los primitivos sistemas de *"hipertexto"* podrían asimilarse a los enlaces de las páginas web actuales.

Tras finalizar el desarrollo de su sistema de *"hipertexto"*, Tim Berners-Lee lo presentó a una convocatoria organizada para desarrollar un sistema de *"hipertexto"* para Internet. Después de unir sus fuerzas con el ingeniero de sistemas Robert Cailliau, presentaron la propuesta ganadora llamada *WorldWideWeb (W3)*.

El primer documento formal con la descripción de HTML se publicó en 1991 bajo el nombre"HTML tags"(*Etiquetas HTML*) y todavía hoy puede ser consultado online a modo de *reliquia informática*.

La primera propuesta oficial para convertir HTML en un estándar se realizó en 1993 por parte del organismo IETF (*Internet Engineering Task Force*). Aunque se consiguieron avances significativos (en esta época se definieron las etiquetas para imágenes, tablas y formularios) ninguna de las dos propuestas de estándar, llamadas HTML y HTML+ consiguieron convertirse en estándar oficial.

En 1995, el organismo IETF organiza un grupo de trabajo de HTML y consigue publicar, el 22 de septiembre de ese mismo año, el estándar HTML 2.0. A pesar de su nombre, HTML 2.0 es el primer estándar oficial de HTML.

A partir de 1996, los estándares de HTML los publica otro organismo de estandarización llamado W3C (*World Wide Web Consortium*). La versión HTML 3.2 se publicó el 14 de Enero de 1997 y es la primera recomendación de HTML publicada por el W3C. Esta revisión incorpora los últimos avances de las páginas web desarrolladas hasta 1996, como *applets* de Java y texto que fluye alrededor de las imágenes.

HTML 4.0 se publicó el 24 de Abril de 1998 (siendo una versión corregida de la publicación original del 18 de Diciembre de 1997) y supone un gran salto desde las versiones anteriores. Entre sus novedades más destacadas se encuentran las hojas de estilos CSS, la posibilidad de incluir pequeños programas o *scripts* en las páginas web, mejora de la accesibilidad de las páginas diseñadas, tablas complejas y mejoras en los formularios.

La última especificación oficial de HTML se publicó el 24 de diciembre de 1999 y se denomina HTML 4.01. Se trata de una revisión y actualización de la versión HTML 4.0, por lo que no incluye novedades significativas.

Desde la publicación de HTML 4.01, la actividad de estandarización de HTML se detuvo y el W3C se centró en el desarrollo del estándar XHTML. Por este motivo, en el año 2004, las empresas Apple, Mozilla y Opera mostraron su preocupación por la falta de interés del W3C en HTML y decidieron organizarse en una nueva asociación llamada WHATWG (*Web Hypertext Application Technology Working Group*).

La actividad actual del WHATWG se centra en el futuro estándar HTML 5, cuyo primer borrador oficial se publicó el 22 de enero de 2008. Debido a la fuerza de las empresas que forman el grupo WHATWG y a la publicación de los borradores de HTML 5.0, en marzo de 2007 elW3C decidió retomar la actividad de estandarizar de HTML.

De forma paralela a su actividad con HTML, W3C ha continuado con la estandarización de XHTML, una versión *avanzada* de HTML y basada en XML. La primera versión de XHTML se denomina XHTML 1.0 y se publicó el 26 de Enero de 2000 (y posteriormente se revisó el 1 de Agosto de 2002).

XHTML 1.0 es una adaptación de HTML 4.01 al lenguaje XML, por lo que mantiene casi todas sus etiquetas y características, pero añade algunas restricciones y elementos propios de XML. La versión XHTML 1.1 ya ha sido publicada en forma de borrador y pretende modularizar XHTML. También ha sido publicado el borrador de

XHTML 2.0, que supondrá un cambio muy importante respecto de las anteriores versiones de XHTML.

4.1.2 Estándares y extensiones

Los desarrolladores de navegadores y aplicaciones relacionadas con HTML siguen las especificaciones y convenios con el fin de que sus productos puedan trabajar con documentos HTML i XHTML válido. Por otro lado los autores crean documentos basándose en los estándares con la finalidad de obtener obras efectivas y correctamente escritas para la visualización en los diversos navegadores. Pero los estándares no son siempre explícitos, los desarrolladores siempre tienen cierta libertad de acción en la forma en la que su software tratará y mostrará la información. Pero estas no son la únicas diferencias que se pueden encontrar entre navegadores, ya que en algunos casos se llegan a añadir extensiones no estandarizadas para mejorar o añadir nuevas funcionalidades.

Un ejemplo del primer caso puede ser el tratamiento del valor que tiene un atributo como el maxlength de un input, en el caso de Firefox y Opera ingnoran el "0" y el calor por defecto es "2" mientras que Internet Explorer ignora el campo entero.

Para otro lado están las extensiones añadidas por terceras personas, para ofrecer funcionalidades y así también poderse diferenciar de los navegadores autorizados. Esto puede llegar a ser un horror para los autores, ya que quieren incluir las ultimas y mas nuevas funcionalidades para estar al dia en su web, pero resulta que no todas están soportadas por los navegadores, o en otro caso se puede encontrar o detectar en que diferentes navegadores pueden tener una manera diferente de hacer las misma cosa o mostrar el mismo contenido.

Estas extensiones pueden llegar a estar tan utilizadas y reconocidas que incluso acaben siendo introducidas como estándar en nuevas y futuras versiones, muchas de estas extensiones han sido introducidas por empresas y corporaciones como pueden ser Sun Microsystems, Netscape o Microsoft. Existen etiquetas como marque, que crean una marquesina para el contenido que engloba, o bgsound para tener sonido de fondo mientras visualizas el documento, que sólo tiene soporte con el navegador de Microsoft, hay otros que han sido adaptados como puede ser la etiqueta <applet>(actualmente obsoleta haciéndose uso de la etiqueta <object>), o la extensión de frames, script, etc... o otras etiquetas i extensiones que ofrece el navegador Netscape(Mozilla en la actualidad) que son algunos claros ejemplos de

extensiones propias, que con el tiempo. la fama y sobretodo el buen funcionamiento de la misma fueron añadidas a las nuevas revisiones de HTML como por ejemplo: frame, javascript, añadiendo nuevos atributos a elementos HTML ya existentes, ampliando sus opciones de visualización.

Así pues te puedes encontrar con la disyuntiva de incorporar nuevas funcionalidades o aplicaciones en tus documentos o bien perder parte de los posibles usuarios que por no utilizar un navegador con soporte para las extensiones no podrá visualizar el documento correctamente o incluso ni tan solo visualizarlo en algunos casos. Si te encuentras en la situación de verte obligado a utilizar estas extensiones no estándares, una posible solución puede pasar por realizar diferentes versiones o alternativas que cumplan con los estándares i que permitan su visualización aun que sea con un comportamiento diferente, suele ser una buena practica.

4.1.3 Etiquetas

En HTML para que los navegadores puedan renderizar los documentos se interpretan una etiquetas que son las que indican la restructuración de cada uno de los elementos del documento, por eso podemos ver un documento HTML como un árbol de etiquetas, ya que cada etiqueta puede contener otras i todas tienen una etiqueta padre de la que provienen.

Estas etiquetas normalmente comienzan y acaban, pero según el elemento del que se trate y de la versión es más o menos restrictivo. La gran diferencia entre el HTML y el XHTML radica en la necesidad obligatoria de tener que cerrar todas las etiquetas, este comportamiento viene dado por el XML y que se ha fusionado para dar lugar al XHTML. Otra diferencias la sensibilidad a las mayúsculas, en el caso del html no discrimina si lo son o no, pero en xhtml las etiquetas tienen que ser escritas en minúsculas.

Las etiquetas HTML están incrustadas dentro del documento, y son las que guían el contenido del documento, y marcan el que se mostrara. Los navegadores utilizan la información dentro de estas etiquetas para decidir como mostrar o bien como tratar el subconjunto de elementos del documento. La primera palabra del tag es el nombre de su formato, normalmente suele ser bastante descriptivo con la función que realiza dentro del documento. Las palabras adicionales que se pueden encontrar dentro de una etiqueta son atributos especiales, algunos pueden tener

asignados unos valores a continuación, que acabaran de definir o modificar la acción del tag.

La mayoría de los tags afectan a una parte en concreto del documento. La región afectada comienza donde por primera vez aparece la etiqueta y sus atributos, y continua hasta que llega a una etiqueta de cierre. Esta etiqueta tiene el mismo nombre que la etiqueta que se ha abierto precedida de "/", además estas etiquetas nunca llevan atributos. En HTML la mayoría de etiquetas tienen un cierre, pero hay algunas que no, como la etiqueta <link> (enlace a un fichero CSS externo). También se tiene en cuenta que el hecho de dejar de añadir según que etiquetas no tiene que contar necesariamente en una incorrecta visualización, los navegadores puede introducir etiquetas obvias alrededor de un texto al interpretar el documento HTML, esto pasa con la etiqueta <p> (tag que define un párrafo).

La estructura del esqueleto de un documento HTML el tag <html></html>. El estándar de html y xhtml por tal que lo cumplan requieren de esta etiqueta, pero en este caso la mayoría de los navegadores pueden detectar y mostrar información codificada en html en documentos que falta la etiqueta. Dentro de los documentos html encontramos dos estructuras principales <head><body>. Dentro de la primera se coloca información sobre el propio documento y dentro de la segunda esta en contenido que queremos enseñar en la ventana del navegador, esta es la que lleva casi toda la carga del documento en la gran mayoría de casos. Otro elemento estructural que el estándar obliga a introducir, pero que otra vez los navegadores no obligan a sus autores a ser introducidos es el tag <title>. El contenido de este se muestra generalmente en la parte superior de la ventana que hace de marco del propio navegador.

Exceptuando las etiquetas comentadas y que están especificadas como obligatorias para los estándares, no hay otros elementos estructurales que sean de insertar dentro de un documento. El resto de elementos para introducir son de libre elección y opcionales. Pese a la gran cantidad de elementos estructurales para introducir, el contenido que se añadirá se puede clasificar en tres grandes categorías: tags (ya comentados), texto y comentarios. Al ver nada más estas tres grandes categorías se puede llegar a pensar erróneamente que se está olvidando un gran grupo, el que contiene los elementos multimedia, pero el caso es que una imagen, sonidos, videos o otros elementos multimedia no son más que referencias insertadas a los documentos hacia ficheros externos, y el navegador es quien se encarga de utilizar estas referencias para cargar y integrar estos contenidos en el propio documento html.

4.1.3.1 Principales etiquetas

La mayoría de las funcionalidades vienen dados por un conjunto de etiquetas de las cuales después dependerán otras y acaban de definir su estructura y propiedades. Con estos subconjuntos de etiquetas se pueden definir la mayoría de etiquetas de elementos que se utilizan y que es común encontrar dentro de los documentos XHTML o HTML.

Las etiquetas de marcaje relacionadas con texto comprenden un gran conjunto dentro de estos lenguajes, esto viene dado por el origen del lenguaje, que se utilizo para poder enriquecerlo, estructurar y organizar mejor el texto y para su orientación académica, para poder ser escaneado y distribuido a través de internet. Por este motivo el diseño se ha considerado secundario respeto a la estructura, ya que aunque nosotros podamos analizar textos y estructurarlos según como se ve, las máquinas acostumbran a leer por elementos marcados.

A la hora de modificar la apariencia del texto tenemos diferentes opciones, los estilos de texto basados en sus contenidos, son etiquetas que pueden modificar la apariencia del texto, pero que además quieren dar a aquella parte del texto un uso o sentido especial. Estas etiquetas pueden ser <cite>, para denominar una cita, , para hacer énfasis, o <var>, para añadir código de programación, dejando de lado el posible cambio a la visualización, puede tener aplicaciones en la búsqueda de contenido a través de estas etiquetas, por eso es importante utilizarlas cuando sea necesario y por su contexto, no por la apariencia que puedan dar.

Otra opción es utilizar los estilos físicos, ésta es una forma de otorgar una apariencia especifica a un texto, sin tener que darle un significado especial. Comúnmente se puede confundir esta practica con la descrita anteriormente, esta opción se tendría que utilizar cuando haya que prevalecer el fondo por encima de las funciones.

HTML también trabaja con unos elementos que son los denominados caracteres especiales, nos podemos encontrar con problemas para introducir ciertos caracteres, como puede ser "<" que pueden confundir al navegador al interpretar el documento, así pues de da una vía alternativa para poder introducir cualquier carácter dentro del conjunto de caracteres Unicode con la codificación de su entidad de carácter. También nos podemos encontrar en situación de colocar o introducir

otro carácter que no es posible introducir vía teclado, como puede ser el símbolo del copyright, la letra pi o una simple flecha, esto y lo descrito anteriormente se puede conseguir con dos formas, en las dos se empieza con "&" i se acaba con ";", en medio se puede utilizar el número de la posición del carácter dentro de la tabla Unicode precedido con "#", o bien directamente con el nombre que se utiliza para la entidad.

La forma en la que se visualiza un documento en pantalla es dejando fluir el texto que contiene, los saltos de línea y espacios que se van encontrandono se tienen en cuenta y no se muestran, el navegador llenará hasta que pueda cada línea de texto dentro de la pantalla hasta el final y siguiendo desde el principio en una nueva línea. Con tal de poder controlar y organizar el texto de una forma mas leíble y estructurada, html ofrece ciertas formas, una de ellas consiste en agrupar el contenido dentro de unos elementos estructurales como pueden ser una división , párrafo <p>, otra opción seria estructurar las partes en secciones con <h1> - <h6>, estos elementos separan y rompen el fluir del texto y lo separa en nuevas líneas. En los primeros se pueden usar con atributos para modificar la visualización del contenido que delimitan además de estructurarlo. Otros elementos comunes serían el salto de línea
, las reglas horizontales <hr>. También está la opción de mostrar el texto tal y como está escrito, con saltos de línea, sangría, espacios,… Además obliga a que aunque la ventana del navegador siga reenderezando el contenido de este bloque no se vea como pasa en otros casos.

Un elemento también muy importante y vital para los documentos HTML son los links, ya hemos comentado alguno de sus usos, como pueden ser las imágenes y cualquier otro medio multimedia, hemos dicho que no son más que links que el propio navegador lee, obtiene e introduce dentro del documento y este elemento es tan importante porque así podemos enlazar cualquier tipo de localización, no obstante, que podamos enlazar no significa que todos los medios que añadamos se pueden integrar dentro de nuestros documentos HTML. En el caso de que no se puedan integrar directamente en el contenido del documento no quiere decir que el navegador no sepa manejarlo, ya que en ocasiones un plugin o un programa externo sea el encargado de obtener el enlace y ejecutarlo.

También tiene otra función muy importante e interesante para los autores de documentos y una vez más es la escritura, según el contenido que tenga un documento puede llegar a ser muy largo y pesado, así pues el hecho de poder enlazar con otros recursos como pueden ser otros documentos HTML nos ofrece la posibilidad de separar contenidos y estructurarlos según nos convenga para hacer

la visualización menos pesada y más agradable, organizada y accesible para los lectores, haciendo uso de los atributos como "id" y "name" se puede hacer que se enlace a un elemento del mismo documento. La forma de introducir estos enlaces a nuestros documentos es mediante la etiqueta <a> (esta etiqueta se llama ancla en ingles, recibió el nombre porque tiene la función de anclar los enlaces y elementos dentro del mismo documento), la parte de texto y imágenes que esté contenida dentro del bloque de la etiqueta será la que activará el enlace si el usuario interactúa con ella y desencadenará la acción de ir al recurso en concreto.

Otro elemento muy común de los documentos HTML son las imágenes, como ya se ha comentado se trata de enlaces que el navegador lee y obtiene la imagen, haciendo uso de los descodificadores con los que ya viene provisto, procesa y la introduce. Las imágenes no sólo se utilizan como ilustraciones sino también como a iconos, puntos clave de listas y otros pequeños elementos decorativos para dar formato al texto. Otra aplicación es como image map, utilizando una imagen como un mapa en el que cuando el lector hace clic en una parte de la imagen se hará una llamada a una url especifica en la que se enviaran las coordenadas del clic del usuario al servidor y este calculara la posición para ejecutar la funcionalidad que tenga asignada el mapa. Actualmente como los autores no tienen acceso al servidor se utiliza mediante javascript, esta tarea del lado del cliente. También se puede combinar con as etiquetas <map> y <area> para no tener que depender del servidor web, introduciendo toda la información en HTML.

Los que se utilizan normalmente para estructurar el contenido son las listas, las hay de tres tipos, ordenadas , desordenadas y de definiciones <dt></dt>. La diferencia entre el primer tipo y el segundo es bien claro, en el caso de las listas de definiciones para cada elemento esta el nombre o el título y en la siguiente línea de la definición correspondiente. El punto fuerte de remarcar de las listas es la capacidad para insertar una gran variedad de elementos de HTML en su interior, incluso se soportan listas dentro de listas.

El elemento para que los usuarios puedan interaccionar dentro del documento son los formularios <form></form>. Dentro de un formulario se pueden introducir una gran cantidad de elementos como entradas para texto, etiquetas, valores por defecto, comboboxes, checkboxes, botones, el formulario interactuará según como este definido su comportamiento, pudiendo enviar la información al propio servidor para que procese ésta, o bien se puede procesar desde el lado del cliente con código javascript entre otros. Una de las funcionalidades importantes de la que no hace gala es la verificación automática de los campos del formulario, esto se puede

salvar haciendo uso de algún script si no se quiere esperar a enviar la información al servidor y evitar volver a cargar para informar del error.

Otro subconjunto de etiquetas para dar formato a nuestra información son las tablas, se pueden definir fila a fila <tr></tr>, con cabeceras <th></th>, o con la información de la tabla <td></td> se tiene que mencionar que hay un número de atributos que permiten adaptar la apariencia, formato y dimensiones. Es habitual utilizar las tablas no sólo para mostrar datos o como típicas tablas, sino también para estructurar el contenido gracias a su variedad en el formato en que se pueden visualizar.

Otro elemento importante a la hora de estructurar el documento HTML son los frames <frame>. Los frames provocan la división de la ventana en ventanas independientes con sus propios contenidos, que no tienen que ser necesariamente otros documentos HTML. así, si un frame contiene enlaces se puede controlar el comportamiento, haciendo que reemplace el contenido del mismo frame, el de otro, o toda la página del navegador. Los frames comunes se tienen que definir dentro de etiquetas <frameset> que en este caso además substituyen la etiqueta <body> y son las encargadas de definir como dividir la ventana del navegador.

Hay otro tipo de frame, muy práctico y usable, se trata del <iframe> inline frame, a diferencia de los otros no necesita esta contenido dentro de un <frameset>, y se comportan como una imagen, definiendo un área rectangular que muestra otro documento pudiendo tenerlos elementos propios de una ventana: scroll, bordes como los otros frames pueden ser introducidos como cualquier imagen dentro de un documento, éste es el caso que utilizaremos nosotros para visualizar alguna de nuestra aplicaciones.

4.1.3.2 Contenidos ejecutables

Una de las capacidades mas útiles que nos podemos encontrar para extender los nuevos documentos con nuevas capacidades/funcionalidades es la de añadir aplicaciones que complementan con nuestro documento. Las formas más comunes para hacerlo es con scripts y con applets. Los applets son pequeñas aplicaciones basadas en java, independientemente de la plataforma. Durante la ejecución estos programas pueden ejecutar contenido dinámicamente en el cliente y en su navegador.

También puede interactuar el usuario, validar información, crear ventanas, ejecutar aplicaciones independientes a la página. Es una forma de poder facilitar aplicaciones y otras funcionalidades al usuario sin que tenga que actualizar su navegador o el software, ya que no se necesita ninguna extensión ni software específico para ejecutarlo, mientras que el navegador utilizado soporte los applet y actualmente todos los navegadores populares los soportan. Los applets se pueden introducir de diferentes maneras dentro de un documento HTML, la primera es haciendo uso de la etiqueta <applet>.

Esta etiqueta actualmente está obsoleta, y ha sido substituida por la etiqueta <object>, la cual se utiliza para introducir dentro del documento applets y otros elementos como podrían ser imágenes y también otro contenido no-HTML/XHTML, como puede ser contenido multimedia. En el caso de utilizar el parámetro data dentro de <object>, el navegador descodificará el tipo de contenido y si puede lo reenderezará directamente, pero en el caso de que no pueda invocará el plugin asociado a la aplicación para que se encargue de ejecutarlo.

La etiqueta <object> fue originalmente implementada por Microsoft para soportar los controles ActiveX, más tarde añadiría soporte para Java, de manera similar, Netscape inicialmente soportaba la alternativa <embid> y <applet> para la inclusión de objetos y más tarde añadirían también soporte para <object>. Aun así hay una resistencia a soportar la etiqueta, por un lado está el perder parte de navegadores que no llegan a ofrecer soporte para etiquetas obsoletas, y por otro lado esta relacionado con quejas sobre una especificación vaga de cómo los navegadores tendrían que implementar el soporte a los applets de Java y el soporte actual de los navegadores es inconsistente, por este motivo algunos como Sun recomiendan que se siga utilizando applets para incluirlos.

En conclusión, los puntos fuertes de un applet frente a una aplicación los encontramos en que éste puede obtener la página del navegador, acceder y reenderezarla HTML, también puede ocasionar la ejecución de código sin conexión, como hemos comentado esta la ventaja de que el usuario no tiene que instalar nada, sino que automáticamente se obtiene la ultima versión del software. Pero también hay puntos negativos: para poder ejecutar funcionalidades importantes dentro del código se tienen que utilizar applets firmados que el usuario tiene que aceptar.

El applet consume sus propios recursos pero tienen que coexistir con el navegador, esto puede parecer que hoy en día no es importante pero en los inicios cuando las

cantidades de memoria RAM no eran tan grandes podía llegar a ser un problema, la falta de recursos, por esto teniendo en cuenta el consumo amplio de banda que puede generar, se tienen que diseñar y utilizar teniendo muy presente como puede afectar a los usuarios, y administrar los recursos para satisfacer a la mayoría de la audiencia (balance entre servidor y cliente). También nos podemos encontrar Bugs en el soporte de algunos navegadores y intentar salvarlos para toda una variedad de navegadores puede ser complicado, no obstante, este es un problema que nos encontramos en la gran mayoría de las tecnologías.

La velocidad a la que se ejecuta un código será ligeramente inferior a la de una aplicación escrita en c++ por ejemplo, ya que no se puede utilizar compiladores de código nativo, sino que esta limitado a la maquina virtual que soporta el navegador. Respeto al control y acceso que tiene un applet, los navegadores asignan una parte del espacio del documento donde se ejecutará el applet.

Como aplicación de seguridad que ofrecería al cliente dependería de la versión que se ejecute si es el caso de un applet sin firmar, su acceso al sistema de ficheros y permisos de la máquina cliente está limitada y también tiene el acceso web limitado al host del applet eso es importante, ya que de esta forma se evita que puedan incluir applets a documentos HTML que cuando sean ejecutados por el cliente provoquen un ataque DOS a una máquina (un ataque a la misma web que contiene el applet es un sin sentido, ya que normalmente los que administran la web, también administran su contenido).

4.1.4 Tipos de contenido

La declaración del DOCTYPE tendría que ser la primera sentencia en aparecer en el documento html antes de una etiqueta <html>. No se considera una etiqueta propiamente de html, es una introducción que le dice al navegador que estemos usando en que versión del lenguaje esta escrito el documento. El DTD define las etiquetas y la sintaxis que se utilizan para crear documentos html.

En caso de asignar un DTD que no corresponda con la versión del lenguaje que se utilizara puede causar que el navegador no entienda correctamente el documento al visualizarlo. Esto se da porque los navegadores pueden renderizar los documentos en dos modos, el modo estándar en el que trabajan según los estándares y el documento se reenderezará como se espera. Pero en el otro modo, quirk que se activa cuando se encuentra con un documento que no cumple con los estándares

especificados, el navegador lo muestra como si lo hiciese un navegador antiguo que no respetaba en todos los casos los entándares .

El caso es que se ha detectado que hay diferencias entre navegadores en modo quirks o muchos propiedades de CSS que se ven afectadas como: padding, margin de las imágenes, el centrado de un bloque con margin: auto no funciona, la herencia de fuentes des de body o tables, la declaración de los tamaños de fuentes en una celda de una tabla es relativo respecto a la fuente del navegador y no la de su elemento padre... son algunos elementos entre otros que pueden afectar gravemente la visualización esperada.

Entre HTML Strict, Transitional y Frameser, en los tres casos el documento XHTML tiene que estar correctamente escrito y en un formato correcto. En el primero de los casos no soporta las etiquetas antiguas y no se permite el uso de framesets, es el mas estricto de los tres. El segundo es como el primero pero sacando la restricción a las etiquetas antiguas, y el ultimo de los tres saca la restricción de los framsets.

4.1.5 Atributos

Otros elementos importantes del HTML son los atributos, se utilizan para extender las propiedades de una etiqueta, cuando el navegador interpreta las etiquetas también busca pares de atributo/valor dentro de cada etiqueta para ver si tiene, y lo que acaba reenderezando es la etiqueta con unas características dadas por estos valores.

Algunas etiquetas tienen atributos por defecto que pueden ser modificados por los que son especificados directamente, como los bordes de las tablas que son nulos por defecto.

Los atributos pueden ser de muchos tipos, pueden afectar a la apariencia de los elementos en los que están definidos como color, el tipo de fuente, también pueden afectar a su comportamiento, como por ejemplo los atributos de eventos: onmouseover, onmouseout, que hacen que si se cumple el evento por el cual están definidos, ejecutarán la función declarada, que puede ser código en un lenguaje client-side como el javascript. También pueden aportar información por definir un elemento, como puede ser el campo "id" o "class", los cuales respectivamente declaran un identificador de un elemento con el valor dado, y añaden una clase al

elemento, además estos elementos también pueden servir para discriminar los estilos que los afectan.

4.2 CSS

4.2.1¿Qué es CSS?

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas.

Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo (también llamados *"documentos semánticos"*). Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para *marcar* los contenidos, es decir, para designar la función de cada elemento dentro de la página: párrafo, titular, texto destacado, tabla, lista de elementos, etc.

Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, etc.

4.2.2 Historia de CSS

Las hojas de estilos aparecieron poco después que el lenguaje de etiquetas SGML, alrededor del año 1970. Desde la creación de SGML, se observó la necesidad de definir un mecanismo que permitiera aplicar de forma consistente diferentes estilos a los documentos electrónicos.

El gran impulso de los lenguajes de hojas de estilos se produjo con el boom de Internet y el crecimiento exponencial del lenguaje HTML para la creación de documentos electrónicos. La guerra de navegadores y la falta de un estándar para

la definición de los estilos dificultaban la creación de documentos con la misma apariencia en diferentes navegadores.

El organismo W3C (World Wide Web Consortium), encargado de crear todos los estándares relacionados con la web, propuso la creación de un lenguaje de hojas de estilos específico para el lenguaje HTML y se presentaron nueve propuestas. Las dos propuestas que se tuvieron en cuenta fueron la CHSS (*Cascading HTML Style Sheets*) y la SSP (*Stream-based Style Sheet Proposal*).

La propuesta CHSS fue realizada por Håkon Wium Lie y SSP fue propuesto por Bert Bos. Entre finales de 1994 y 1995 Lie y Bos se unieron para definir un nuevo lenguaje que tomaba lo mejor de cada propuesta y lo llamaron CSS (*Cascading Style Sheets*).

En 1995, el W3C decidió apostar por el desarrollo y estandarización de CSS y lo añadió a su grupo de trabajo de HTML. A finales de 1996, el W3C publicó la primera recomendación oficial, conocida como "CSS nivel 1".

A principios de 1997, el W3C decide separar los trabajos del grupo de HTML en tres secciones: el grupo de trabajo de HTML, el grupo de trabajo de DOM y el grupo de trabajo de CSS.

El 12 de Mayo de 1998, el grupo de trabajo de CSS publica su segunda recomendación oficial, conocida como "CSS nivel 2". La versión de CSS que utilizan todos los navegadores de hoy en día es CSS 2.1, una revisión de CSS 2 que aún se está elaborando (la última actualización es del 23 de abril de 2009). Al mismo tiempo, la siguiente recomendación de CSS, conocida como "CSS nivel 3", continúa en desarrollo desde 1998 y hasta el momento sólo se han publicado borradores.

La adopción de CSS por parte de los navegadores ha requerido un largo periodo de tiempo. El mismo año que se publicó CSS 1, Microsoft lanzaba su navegador Internet Explorer 3.0, que disponía de un soporte bastante reducido de CSS. El primer navegador con soporte completo de CSS 1 fue la versión para Mac de Internet Explorer 5, que se publicó en el año 2000. Por el momento, ningún navegador tiene soporte completo de CSS 2.1.

4.2.3 Soporte de CSS en los navegadores

El trabajo del diseñador web siempre está limitado por las posibilidades de los navegadores que utilizan los usuarios para acceder a sus páginas. Por este motivo es imprescindible conocer el soporte de CSS en cada uno de los navegadores más utilizados del mercado.

Internamente los navegadores están divididos en varios componentes. La parte del navegador que se encarga de interpretar el código HTML y CSS para mostrar las páginas se denomina motor. Desde el punto de vista del diseñador CSS, la versión de un motor es mucho más importante que la versión del propio navegador.

La siguiente tabla muestra el soporte de CSS 1, CSS 2.1 y CSS 3 de los cinco navegadores más utilizados por los usuarios:

Navegador	Motor	CSS 1	CSS 2.1	CSS 3
Internet Explorer	Trident	Completo desde la versión 6.0	Completo desde la versión 8.0	Prácticamente nulo
Firefox	Gecko	Completo	Casi completo	Selectores, pseudo-clases y algunas propiedades
Safari	WebKit	Completo	Casi completo	Todos los selectores, pseudo-clases y muchas propiedades
Opera	Presto	Completo	Casi completo	Todos los selectores, pseudo-clases y muchas propiedades
Google Chrome	WebKit	Completo	Casi completo	Todos los selectores, pseudo-clases y muchas propiedades

Los navegadores Safari y Opera son los más avanzados en el soporte de CSS, ya que incluyen muchos elementos de la futura versión CSS 3 y un soporte casi perfecto de la actual version 2.1. El navegador Firefox no tiene un soporte tan avanzado de CSS 3 pero las últimas versiones están alcanzando rápidamente a Safari y Opera.

Por su parte, el navegador Internet Explorer sólo puede considerarse adecuado desde el punto de vista de CSS a partir de su versión 7. Internet Explorer 6,

utilizado todavía por un número significativo de usuarios, sufre carencias muy importantes y contiene decenas de errores en su soporte de CSS. Internet Explorer 8 incluye el soporte completo de todas las propiedades y características de CSS 2.1.

La tabla anterior ha sido elaborada a partir de la información que se puede encontrar en la página *Comparison of layout engines* de la Wikipedia, donde se muestra una comparación exhaustiva sobre el soporte de todas las características de CSS por parte de cada navegador.

4.2.4 Especificación oficial

La especificación o norma oficial que se utiliza actualmente para diseñar páginas web con CSS es la versión CSS 2.1, actualizada por última vez el 23 de abril de 2009 y que se puede consultar libremente en http://www.w3.org/TR/CSS21/

Desde hace varios años, el organismo W3C trabaja en la elaboración de la próxima versión de CSS, conocida como CSS 3. Esta nueva versión incluye multitud de cambios importantes en todos los niveles y es mucho más avanzada y compleja que CSS 2.

No obstante, pasarán muchos años hasta que se publique la versión definitiva completa de CSS 3 y hasta que los principales navegadores del mercado incluyan la mayor parte del nuevo estándar.

El sitio web del organismo W3C dispone de una sección en la que se detalla el trabajo que el W3C está desarrollando actualmente en relación a CSS y también dispone de un blog en el que se publican todas las novedades relacionadas con CSS.

4.2.5 Funcionamiento basico

Antes de que se generalizara el uso de CSS, los diseñadores de páginas web utilizaban etiquetas HTML especiales para modificar el aspecto de los elementos de la página. El siguiente ejemplo muestra una página HTML con estilos definidos sin utilizar CSS:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<metahttp-equiv="Content-Type"content="text/html; charset=iso-8859-1" />
<title>Ejemplo de estilos sin CSS</title>
</head>

<body>
<h1><fontcolor="red"face="Arial"size="5">Titular de la página</font></h1>
<p><fontcolor="gray"face="Verdana"size="2">Un párrafo de texto no muy largo.</font></p>
</body>
</html>
```

El ejemplo anterior utiliza la etiqueta `<font>` con sus atributos `color`, `face` y `size` para definir el color, el tipo y el tamaño de letra de cada elemento de la página.

El problema de utilizar este método para definir el aspecto de los elementos se puede ver claramente con el siguiente ejemplo: si la página tuviera 50 elementos diferentes, habría que insertar 50 etiquetas `<font>`. Si el sitio web entero se compone de 10.000 páginas diferentes, habría que definir 500.000 etiquetas `<font>`. Como cada etiqueta `<font>` tiene tres atributos, habría que definir 1.5 millones de atributos.

Como el diseño de los sitios web está en constante evolución, es habitual modificar cada cierto tiempo el aspecto de las páginas del sitio. Siguiendo con el ejemplo anterior, cambiar el aspecto del sitio requeriría modificar 500.000 etiquetas y 1.5 millones de atributos.

La solución que propone CSS es mucho mejor, como se puede ver en el siguiente ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<metahttp-equiv="Content-Type"content="text/html; charset=iso-8859-1" />
<title>Ejemplo de estilos con CSS</title>
<styletype="text/css">
  h1 { color: red;  font-family: Arial;   font-size: large;  }
  p  { color: gray; font-family: Verdana; font-size: medium; }
</style>
```

```
</head>

<body>
<h1>Titular de la página</h1>
<p>Un párrafo de texto no muy largo.</p>
</body>
</html>
```

CSS permite separar los contenidos de la página y la información sobre su aspecto. En el ejemplo anterior, dentro de la propia página HTML se crea una zona especial en la que se incluye toda la información relacionada con los estilos de la página.

Utilizando CSS, se pueden establecer los mismos estilos con menos esfuerzo y sin *ensuciar* el código HTML de los contenidos con etiquetas `<font>`. Como se verá más adelante, la etiqueta `<style>` crea una zona especial donde se incluyen todas las reglas CSS que se aplican en la página.

En el ejemplo anterior, dentro de la zona de CSS se indica que todas las etiquetas `<h1>` de la página se deben ver de color rojo, con un tipo de letra Arial y con un tamaño de letra grande. Además, las etiquetas `<p>` de la página se deben ver de color gris, con un tipo de letra Verdana y con un tamaño de letra medio.

Definir los estilos de esta forma ahorra miles de etiquetas y millones de atributos respecto a la solución anterior, pero sigue sin ser una solución ideal. Como los estilos CSS sólo se aplican en la página que los incluye, si queremos que las 10.000 páginas diferentes del sitio tengan el mismo aspecto, se deberían copiar 10.000 veces esas mismas reglas CSS. Más adelante se explica la solución que propone CSS para evitar este problema.

4.3 Javascript

4.3.1 ¿Qué es JavaScript?

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas.

Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario.

Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.

A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. Legalmente, JavaScript es una marca registrada de la empresa Sun Microsystems, como se puede ver en http://www.sun.com/suntrademarks/.

4.3.2 Historia

A principios de los años 90, la mayoría de usuarios que se conectaban a Internet lo hacían con módems a una velocidad máxima de 28.8 kbps. En esa época, empezaban a desarrollarse las primeras aplicaciones web y por tanto, las páginas web comenzaban a incluir formularios complejos.

Con unas aplicaciones web cada vez más complejas y una velocidad de navegación tan lenta, surgió la necesidad de un lenguaje de programación que se ejecutara en el navegador del usuario. De esta forma, si el usuario no rellenaba correctamente un formulario, no se le hacía esperar mucho tiempo hasta que el servidor volviera a mostrar el formulario indicando los errores existentes.

Brendan Eich, un programador que trabajaba en Netscape, pensó que podría solucionar este problema adaptando otras tecnologías existentes (como *ScriptEase*) al navegador Netscape Navigator 2.0, que iba a lanzarse en 1995. Inicialmente, Eich denominó a su lenguaje *LiveScript*.

Posteriormente, Netscape firmó una alianza con Sun Microsystems para el desarrollo del nuevo lenguaje de programación. Además, justo antes del lanzamiento Netscape decidió cambiar el nombre por el de JavaScript. La razón del cambio de nombre fue exclusivamente por marketing, ya que Java era la palabra de moda en el mundo informático y de Internet de la época.

La primera versión de JavaScript fue un completo éxito y Netscape Navigator 3.0 ya incorporaba la siguiente versión del lenguaje, la versión 1.1. Al mismo tiempo, Microsoft lanzó JScript con su navegador Internet Explorer 3. JScript era una copia de JavaScript al que le cambiaron el nombre para evitar problemas legales.

Para evitar una guerra de tecnologías, Netscape decidió que lo mejor sería estandarizar el lenguaje JavaScript. De esta forma, en 1997 se envió la especificación JavaScript 1.1 al organismo ECMA (*European Computer Manufacturers Association*).

ECMA creó el comité TC39 con el objetivo de *"estandarizar de un lenguaje de script multiplataforma e independiente de cualquier empresa"*. El primer estándar que creó el comité TC39 se denominó **ECMA-262**, en el que se definió por primera vez el lenguaje ECMAScript.

Por este motivo, algunos programadores prefieren la denominación *ECMAScript* para referirse al lenguaje JavaScript. De hecho, JavaScript no es más que la implementación que realizó la empresa Netscape del estándar ECMAScript.

La organización internacional para la estandarización (ISO) adoptó el estándar ECMA-262 a través de su comisión IEC, dando lugar al estándar ISO/IEC-16262.

4.3.3 Estandares y especificaciones oficiales

ECMA ha publicado varios estándares relacionados con ECMAScript. En Junio de 1997 se publicó la primera edición del estándar ECMA-262. Un año después, en Junio de 1998 se realizaron pequeñas modificaciones para adaptarlo al estandar ISO/IEC-16262 y se creó la segunda edición.

La tercera edición del estándar ECMA-262 (publicada en Diciembre de 1999) es la versión que utilizan los navegadores actuales y se puede consultar gratuitamente en http://www.ecma-international.org/publications/standards/Ecma-262.htm

Actualmente se encuentra en desarrollo la cuarta versión de ECMA-262, que podría incluir novedades como paquetes, *namespaces*, definición explícita de clases, etc.

ECMA también ha definido varios estándares relacionados con ECMAScript, como el estándar ECMA-357, que define una extensión conocida como E4X y que permite la integración de JavaScript y XML.

4.3.4 Como incluir Javascript en un documento XHTML

La integración de JavaScript y XHTML es muy flexible, ya que existen al menos tres formas para incluir código JavaScript en las páginas web.

4.3.4.1 Incluir JavaScript en el mismo documento XHTML

El código JavaScript se encierra entre etiquetas `<script>` y se incluye en cualquier parte del documento. Aunque es correcto incluir cualquier bloque de código en cualquier zona de la página, se recomienda definir el código JavaScript dentro de la cabecera del documento (dentro de la etiqueta `<head>`):

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<metahttp-equiv="Content-Type"content="text/html; charset=iso-8859-1" />
<title>Ejemplo de código JavaScript en el propio documento</title>
<scripttype="text/javascript">
  alert("Un mensaje de prueba");
</script>
</head>

<body>
<p>Un párrafo de texto.</p>
</body>
</html>
```

Para que la página XHTML resultante sea válida, es necesario añadir el atributo `type` a la etiqueta `<script>`. Los valores que se incluyen en el atributo `type` están estandarizados y para el caso de JavaScript, el valor correcto es `text/javascript`.

Este método se emplea cuando se define un bloque pequeño de código o cuando se quieren incluir instrucciones específicas en un determinado documento HTML que completen las instrucciones y funciones que se incluyen por defecto en todos los documentos del sitio web.

El principal inconveniente es que si se quiere hacer una modificación en el bloque de código, es necesario modificar todas las páginas que incluyen ese mismo bloque de código JavaScript.

4.3.4.2 Definir JavaScript en un archivo externo

Las instrucciones JavaScript se pueden incluir en un archivo externo de tipo JavaScript que los documentos XHTML enlazan mediante la etiqueta `<script>`. Se pueden crear todos los archivos JavaScript que sean necesarios y cada documento XHTML puede enlazar tantos archivos JavaScript como necesite.

Ejemplo:

Archivo `codigo.js`

```
alert("Un mensaje de prueba");
```

Documento XHTML

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<metahttp-equiv="Content-Type"content="text/html; charset=iso-8859-1" />
<title>Ejemplo de código JavaScript en el propio documento</title>
<scripttype="text/javascript"src="/js/codigo.js"></script>
</head>

<body>
<p>Un párrafo de texto.</p>
</body>
</html>
```

Además del atributo `type`, este método requiere definir el atributo `src`, que es el que indica la URL correspondiente al archivo JavaScript que se quiere enlazar. Cada

etiqueta <script> solamente puede enlazar un único archivo, pero en una misma página se pueden incluir tantas etiquetas <script> como sean necesarias.

Los archivos de tipo JavaScript son documentos normales de texto con la extensión .js, que se pueden crear con cualquier editor de texto como Notepad, Wordpad, EmEditor, UltraEdit, Vi, etc.

La principal ventaja de enlazar un archivo JavaScript externo es que se simplifica el código XHTML de la página, que se puede reutilizar el mismo código JavaScript en todas las páginas del sitio web y que cualquier modificación realizada en el archivo JavaScript se ve reflejada inmediatamente en todas las páginas XHTML que lo enlazan.

4.3.4.3 Incluir JavaScript en los elementos XHTML

Este último método es el menos utilizado, ya que consiste en incluir trozos de JavaScript dentro del código XHTML de la página:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<metahttp-equiv="Content-Type"content="text/html; charset=iso-8859-1" />
<title>Ejemplo de código JavaScript en el propio documento</title>
</head>

<body>
<ponclick="alert('Un mensaje de prueba')">Un párrafo de texto.</p>
</body>
</html>
```

El mayor inconveniente de este método es que *ensucia* innecesariamente el código XHTML de la página y complica el mantenimiento del código JavaScript. En general, este método sólo se utiliza para definir algunos eventos y en algunos otros casos especiales, como se verá más adelante.

4.3.5 Posibilidades y limitaciones

Desde su aparición, JavaScript siempre fue utilizado de forma masiva por la mayoría de sitios de Internet. La aparición de Flash dismin

uyó su popularidad, ya que Flash permitía realizar algunas acciones imposibles de llevar a cabo mediante JavaScript.

Sin embargo, la aparición de las aplicaciones AJAX programadas con JavaScript le ha devuelto una popularidad sin igual dentro de los lenguajes de programación web.

En cuanto a las limitaciones, JavaScript fue diseñado de forma que se ejecutara en un entorno muy limitado que permitiera a los usuarios confiar en la ejecución de los scripts.

De esta forma, los scripts de JavaScript no pueden comunicarse con recursos que no pertenezcan al mismo dominio desde el que se descargó el script. Los scripts tampoco pueden cerrar ventanas que no hayan abierto esos mismos scripts. Las ventanas que se crean no pueden ser demasiado pequeñas ni demasiado grandes ni colocarse fuera de la vista del usuario (aunque los detalles concretos dependen de cada navegador).

Además, los scripts no pueden acceder a los archivos del ordenador del usuario (ni en modo lectura ni en modo escritura) y tampoco pueden leer o modificar las preferencias del navegador.

Por último, si la ejecución de un script dura demasiado tiempo (por ejemplo por un error de programación) el navegador informa al usuario de que un script está consumiendo demasiados recursos y le da la posibilidad de detener su ejecución.

A pesar de todo, existen alternativas para poder saltarse algunas de las limitaciones anteriores. La alternativa más utilizada y conocida consiste en firmar digitalmente el script y solicitar al usuario el permiso para realizar esas acciones.

4.3.6 JavaScript y navegadores

Los navegadores más modernos disponibles actualmente incluyen soporte de JavaScript hasta la versión correspondiente a la tercera edición del estándar ECMA-262.

La mayor diferencia reside en el *dialecto* utilizado, ya que mientras Internet Explorer utiliza JScript, el resto de navegadores (Firefox, Opera, Safari, Konqueror) utilizan JavaScript.

4.4 PHP

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

4.4.1 Visión general

PHP es un acrónimo recursivo que significa ***P****HP* ***H****ypertext* ***P****re-processor* (inicialmente PHP Tools, o, *P*ersonal *H*ome *P*age Tools). Fue creado originalmente por Rasmus Lerdorf en 1994; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la Free Software Foundation considera esta licencia como software libre.

Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores, el número de sitios en PHP ha compartido algo de su preponderante sitio con otros nuevos lenguajes no tan poderosos desde agosto de 2005. Este mismo sitio web de Wikipedia está desarrollado en PHP. Es también el módulo Apache más popular entre las computadoras que utilizan Apache como servidor web. La versión más reciente de PHP es la 5.3.4, del 10 de diciembre de 2010.

El gran parecido que posee PHP con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy corta. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones.

Aunque todo en su diseño está orientado a facilitar la creación de página web, es posible crear aplicaciones con una interfaz gráfica para el usuario, utilizando la extensión PHP-Qt o PHP-GTK. También puede ser usado desde la línea de órdenes, de la misma manera como Perl o Python pueden hacerlo; a esta versión de PHP se la llama PHP-CLI (*Command Line Interface*).

Cuando el cliente hace una petición al servidor para que le envíe una página web, el servidor ejecuta el intérprete de PHP. Éste procesa el script solicitado que generará el contenido de manera dinámica (por ejemplo obteniendo información de una base de datos). El resultado es enviado por el intérprete al servidor, quien a su vez se lo envía al cliente. Mediante extensiones es también posible la generación de archivos PDF, Flash, así como imágenes en diferentes formatos.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite.

XAMPP es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor Web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl. El programa está liberado bajo la licencia GNU y actúa como un servidor Web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP esta disponible para Microsoft Windows, GNU/Linux, Solaris, y MacOS X.

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como UNIX (y de ese tipo, como Linux o Mac OS X) y Windows, y puede interactuar con los servidores de web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

PHP es una alternativa a las tecnologías de MicrosoftASP y ASP.NET (que utiliza C#VB.NET como lenguajes), a ColdFusion de la compañía Adobe (antes Macromedia), a JSP/Java de Oracle, y a CGI/Perl. Aunque su creación y desarrollo se da en el ámbito de los sistemas libres, bajo la licencia GNU, existe además un IDE (entorno de desarrollo integrado) comercial llamado Zend Studio.

Recientemente, CodeGear (la división de lenguajes de programación de Borland) ha sacado al mercado un entorno integrado de desarrollo para PHP, denominado **Delphi for PHP**. También existen al menos un par de módulos1 para Eclipse, uno de los IDE más populares.

4.4.2 Historia

Fue originalmente diseñado en Perl, con base en la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador danés-canadiense Rasmus Lerdorf en el año 1994 para mostrar su currículum vítae y guardar ciertos datos, como la cantidad de tráfico que su página web recibía. El 8 de junio de 1995 fue publicado "**P**ersonal **H**ome **P**age Tools" después de que Lerdorf lo combinara con su propio *Form Interpreter* para crear PHP/FI.

PHP 3

Dos programadores israelíes del Technion, Zeev Suraski y Andi Gutmans, reescribieron el analizador sintáctico (*parser* en inglés) en el año 1997 y crearon la base del PHP3, cambiando el nombre del lenguaje a la forma actual. Inmediatamente comenzaron experimentaciones públicas de PHP3 y fue publicado oficialmente en junio del 1998.

Para 1999, Suraski y Gutmans reescribieron el código de PHP, produciendo lo que hoy se conoce como motor Zend. También fundaron Zend Technologies en Ramat Gan, Israel.

PHP 4

En mayo de 2000 PHP 4 fue lanzado bajo el poder del motor Zend Engine 1.0. El día 13 de julio de 2007 se anunció la suspensión del soporte y desarrollo de la versión 4 de PHP,2 a pesar de lo anunciado se ha liberado una nueva versión con mejoras de seguridad, la 4.4.8 publicada el 13 de enero del 2008 y posteriormente la versión 4.4.9 publicada el 7 de agosto de 2008.3 Según esta noticia [3] se dará soporte a fallos críticos hasta el 2008-08-09,.....

PHP 5

El 13 de julio de 2004, fue lanzado PHP 5, utilizando el motor Zend Engine 2.0 (o Zend Engine 2). La versión más reciente de PHP es la 5.3.4 (10 de diciembre de 2010), que incluye todas las ventajas que provee el nuevo Zend Engine 2 como:

- Mejor soporte para la Programación Orientada a Objetos, que en versiones anteriores era extremadamente rudimentario.
- Mejoras de rendimiento.
- Mejor soporte para MySQL con extensión completamente reescrita.
- Mejor soporte a XML (XPath, DOM, etc.).
- Soporte nativo para SQLite.
- Soporte integrado para SOAP.
- Iteradores de datos.
- Manejo de excepciones.
- Mejoras con la implementación con Oracle.

Aún se siguen publicando versiones de la rama 5.2.X, siendo publicada la versión 5.2.14 el 22 de Julio de 2010, aunque la mayoría son actualizaciones de seguridad

PHP 6

Está previsto el lanzamiento en breve de la rama 6 de PHP. Cuando se lance esta nueva versión quedarán solo dos ramas activas en desarrollo (PHP 5 y 6), pues se abandonó el desarrollo y soporte de PHP 4 el 13 de julio de 2007.4

Las diferencias que encontraremos frente a PHP 5 son:

- Soportará Unicode;
- limpieza de funcionalidades obsoletas como *register_globals*, *safe_mode*, etc;
- PECL y eliminación de soporte ereg;
- mejoras en orientación a objetos;
- inclusión en el núcleo de *xmlReader* y *xmlWriter* así como *Fileinfo*;
- return por referencia devolverá un error;
- se retira el soporte de las bibliotecas *FreeType1* y *GD1*;
- etc.

Tabla resumen

Versión	Fecha	Cambios más importantes
PHP 1.0	8 de junio de 1995	Oficialmente llamado "Herramientas personales de trabajo (PHP Tools)". Es el primer uso del nombre "PHP".
PHP Version 2 (PHP/FI)	16 de abril de 1996	Considerado por el creador como la "más rápida y simple herramienta" para la creación de páginas webs dinámicas .
PHP 3.0	6 de junio de 1998	Desarrollo movido de una persona a muchos desarrolladores. Zeev Suraski y Andi Gutmans reescriben la base para esta versión.
PHP 4.0	22 de mayo de 2000	Se añade un sistema más avanzado de análisis de etiquetas en dos fases análisis/ejecución llamado el motor Zend.
PHP 4.1	10 de diciembre de 2001	Introducidas las variables superglobals ($_GET, $_SESSION, etc.).
PHP 4.2	22 de abril de 2002	Se deshabilitan register_globals por defecto.
PHP 4.3	27 de diciembre de 2002	Introducido la CLI, en adición a la CGI.
PHP 4.4	11 de julio de 2005	
PHP 5.0	13 de julio de 2004	Motor Zend II con un nuevo modelo de objetos.
PHP 5.1	25 de noviembre de 2005	
PHP 5.2	2 de noviembre de 2006	Habilitado el filtro de extensiones por defecto.
PHP 5.2.4	30 de agosto de 2007	

PHP 5.2.5	8 de noviembre de 2007	Versión centrada en mejorar la estabilidad (+60 errores solucionados).
PHP 5.2.8	8 de diciembre de 2008	
PHP 5.2.9	26 de febrero de 2009	Diversas mejoras en el ámbito de la seguridad (+50 errores solucionados).
PHP 5.2.12	17 de diciembre de 2009	Diversas mejoras en el ámbito de la seguridad (+50 errores solucionados).
PHP 5.3	30 de junio de 2009	namespaces, late static binding, closures, optional garbage collection for cyclic references, nuevas extensiones (+140 errores solucionados).
PHP 5.3.1	19 de noviembre de 2009	Diversas mejoras en el ámbito de la seguridad (36 errores solucionados).
PHP 5.3.2	4 de marzo del 2010	Diversas mejoras en el ámbito de la seguridad (99 errores solucionados).
PHP 5.3.3	22 de julio del 2010	Diversas mejoras en el ámbito de la seguridad y estabilidad. (más de 100 errores solucionados).
PHP 5.3.4	10 de diciembre del 2010	Diversas mejoras en el ámbito de la seguridad y estabilidad. (105 errores solucionados).
PHP 6	S/D	

4.4.3 ¿Cuáles son los beneficios de PHP?

- Es un lenguaje multiplataforma.
- Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una Base de Datos.
- El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial ([4]), entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Tiene manejo de excepciones (desde PHP5).

Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar (muchos otros lenguajes tampoco lo hacen), aun estando dirigido a alguna en particular, el programador puede aplicar en su trabajo cualquier técnica de programación y/o desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del patrón de diseñoModelo Vista Controlador (o MVC), que permiten separar el tratamiento y acceso a los datos, la lógica de control y la interfaz de usuario en tres componentes independientes (ver más abajo Frameworks en PHP).

4.4.4 Inconvenientes

Como es un lenguaje que se interpreta en ejecución para ciertos usos puede resultar un inconveniente que el código fuente no pueda ser ocultado. La ofuscación es una técnica que puede dificultar la lectura del código pero no la impide y aparte en ciertos casos representa un costo en tiempos de ejecución.

4.4.5 Pear: Estándares de desarrollo para php

Pear es el acrónimo de PHP extensión Application Repository. Es parte del proyecto PHP y consiste en una biblioteca de extensión programada en lenguaje PHP que permite desarrollar aplicaciones, módulos y extensiones de alto nivel y calidad.

Por aparte, entre las especificaciones a los programadores, tanto desarrolladores del proyecto PEAR, como programadores en lenguaje PHP, existe una guía de estándares de desarrollo y programación que define el modo en que, básicamente, un script PHP debe ser realizado.

Estándar 1: La indentación debe ser a cuatro espacios sin caracteres de tabulación. Esto es debido a que ciertos IDE's de desarrollo introducen caracteres de tabulación cuando indentan un texto automáticamente. Se recomienda el uso de herramientas o editores generales como EMACS u otros.

Estándar 2: Las estructuras de control deben tener un espacio entre el **keyword** de la estructura y el signo de apertura de paréntesis para distinguir entre las llamadas de las funciones y el signo de llaves debe estar sobre la línea de la estructura.

Estándar 3: Las funciones deben ser llamadas sin espacios entre el nombre de la función, el signo de paréntesis y el primer parámetro; espacios entre cada coma por parámetro y sin espacios entre el ultimo paréntesis, el signo de paréntesis cerrado y el signo de punto y coma (**;**).

Estándar 4: El estilo de los comentarios debe ser como el estilo de comentarios para C (/* */ ó //), no debe de utilizarse el estilo de comentarios de Perl (#).

Estándar 5: Cuando se incluya un archivo de dependencia incondicionalmente utilice **require_once** y cuando sea condicionalmente, utilice **include_once**.

Estándar 6: siempre utilice las etiquetas <?php ?> para abrir un bloque de código. No utilice el método de etiquetas cortas, por que esto depende de las directivas de configuración en el archivo PHP.INI y hace que el script no sea tan portable.

Estándar 7: Los nombres de las clases deben de iniciar con letra mayúscula. Los nombres de las variables y de las funciones pueden iniciar con letra minúscula, pero si estas tienen más de una palabra, cada nueva palabra debe iniciar con letra mayúscula (el nombre puede escribirse separado por signos de guión mayor). Si una función, en una clase, es privada; deberá comenzar con el signo de guión mayor para una fácil identificación. Las constantes deben de escribirse siempre en mayúsculas y tanto estas como las variables globales deben de tener como prefijo el nombre de la clase a la que pertenecen.

Estándar 8: Los archivos con código PHP, deben de ser guardados en formato ASCII utilizando la codificación ISO-8859-1. **(Actualizado)**. El formato ASCII con codificación ISO-8859-1 es el formato en que se guardan los archivos de texto plano (.txt). La razón de este estándar es que determinados editores HTML (en especial Dreamweaver), agregan códigos de carácter extraño de salto de línea (como si se tratara de un archivo binario) y esto puede ocasionar que el interprete de PHP, encuentre problemas a la hora de leer el script.

4.5 SQL

4.5.1 ¿Que es SQL?

Las aplicaciones en red son cada día más numerosas y versátiles. En muchos casos, el esquema básico de operación es una serie de scripts que rigen el comportamiento de una base de datos.

Debido a la diversidad de lenguajes y de bases de datos existentes, la manera de comunicar entre unos y otras sería realmente complicada a gestionar de no ser por la existencia de estándares que nos permiten el realizar las operaciones básicas de una forma universal.

Es de eso de lo que trata el Structured Query Language que no es más que un lenguaje estándar de comunicación con bases de datos. Hablamos por tanto de un lenguaje normalizado que nos permite trabajar con cualquier tipo de lenguaje (ASP

o PHP) en combinación con cualquier tipo de base de datos (MS Access, SQL Server, MySQL...).

El hecho de que sea estándar no quiere decir que sea idéntico para cada base de datos. En efecto, determinadas bases de datos implementan funciones específicas que no tienen necesariamente que funcionar en otras.

Aparte de esta universalidad, el SQL posee otras dos características muy apreciadas. Por una parte, presenta una potencia y versatilidad notables que contrasta, por otra, con su accesibilidad de aprendizaje.

4.5.2 Tipos de datos

Como sabemos una base de datos esta compuesta de tablas donde almacenamos registros catalogados en función de distintos campos (características).

Un aspecto previo a considerar es la naturaleza de los valores que introducimos en esos campos. Dado que una base de datos trabaja con todo tipo de informaciones, es importante especificarle qué tipo de valor le estamos introduciendo de manera a, por un lado, facilitar la búsqueda posteriormente y por otro, optimizar los recursos de memoria.

Cada base de datos introduce tipos de valores de campo que no necesariamente están presentes en otras. Sin embargo, existe un conjunto de tipos que están representados en la totalidad de estas bases. Estos tipos comunes son los siguientes:

Alfanuméricos	Contienen cifras y letras. Presentan una longitud limitada (255 caracteres)
Numéricos	Existen de varios tipos, principalmente, enteros (sin decimales) y reales (con decimales).
Booleanos	Poseen dos formas: Verdadero y falso (Sí o No)
Fechas	Almacenan fechas facilitando posteriormente su explotación. Almacenar fechas de esta forma posibilita ordenar los registros por fechas o calcular los días entre

	una fecha y otra...
Memos	Son campos alfanuméricos de longitud ilimitada. Presentan el inconveniente de no poder ser indexados (veremos más adelante lo que esto quiere decir).
Autoincrementables	Son campos numéricos enteros que incrementan en una unidad su valor para cada registro incorporado. Su utilidad resulta más que evidente: Servir de identificador ya que resultan exclusivos de un registro.

4.5.3 Tipos de datos

Los tipos de datos SQL se clasifican en 13 tipos de datos primarios y de varios sinónimos válidos reconocidos por dichos tipos de datos. Los tipos de datos primarios son:

Tipo de Datos	**Longitud**	**Descripción**
BINARY	1 byte	Para consultas sobre tabla adjunta de productos de bases de datos que definen un tipo de datos Binario.
BIT	1 byte	Valores Si/No ó True/False
BYTE	1 byte	Un valor entero entre 0 y 255.
COUNTER	4 bytes	Un número incrementado automáticamente (de tipo Long)
CURRENCY	8 bytes	Un entero escalable entre 922.337.203.685.477,5808 y 922.337.203.685.477,5807.
DATETIME	8 bytes	Un valor de fecha u hora entre los años 100 y 9999.

SINGLE	4 bytes	Un valor en punto flotante de precisión simple con un rango de - 3.402823*1038 a -1.401298*10-45 para valores negativos, 1.401298*10- 45 a 3.402823*1038 para valores positivos, y 0.
DOUBLE	8 bytes	Un valor en punto flotante de doble precisión con un rango de - 1.79769313486232*10308 a -4.94065645841247*10-324 para valores negativos, 4.94065645841247*10-324 a 1.79769313486232*10308 para valores positivos, y 0.
SHORT	2 bytes	Un entero corto entre -32,768 y 32,767.
LONG	4 bytes	Un entero largo entre -2,147,483,648 y 2,147,483,647.
LONGTEXT	1 byte por carácter	De cero a un máximo de 1.2 gigabytes.
LONGBINARY	Según se necesite	De cero 1 gigabyte. Utilizado para objetos OLE.
TEXT	1 byte por carácter	De cero a 255 caracteres.

La siguiente tabla recoge los sinónimos de los tipos de datos definidos:

Tipo de Dato	**Sinónimos**
BINARY	VARBINARY
BIT	BOOLEAN LOGICAL LOGICAL1 YESNO
BYTE	INTEGER1
COUNTER	AUTOINCREMENT

CURRENCY	MONEY
DATETIME	DATE TIME TIMESTAMP
SINGLE	FLOAT4 IEEESINGLE REAL
DOUBLE	FLOAT FLOAT8 IEEEDOUBLE NUMBER NUMERIC
SHORT	INTEGER2 SMALLINT
LONG	INT INTEGER INTEGER4
LONGBINARY	GENERAL OLEOBJECT
LONGTEXT	LONGCHAR MEMO NOTE
TEXT	ALPHANUMERIC CHAR - CHARACTER STRING - VARCHAR
VARIANT (No Admitido)	VALUE

4.5.4 Tipos de sentencias

En SQL tenemos bastantes sentencias que se pueden utilizar para realizar diversas tareas.

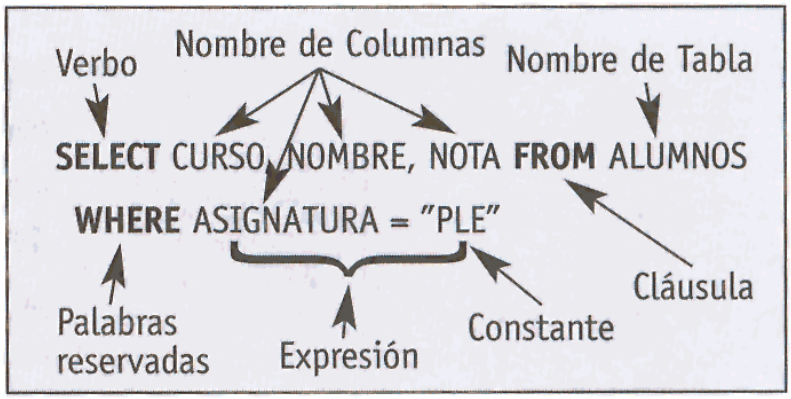
Dependiendo de las tareas, estas sentencias se pueden clasificar en tres grupos principales (DML, DDL,DCL), aunque nos quedaría otro grupo que a mi entender no está dentro del lenguaje SQL sino del PLSQL.

SENTENCIA		DESCRIPCIÓN
DML	**Manipulación de datos** SELECT INSERT DELETE UPDATE	Recupera datos de la base de datos. Añade nuevas filas de datos a la base de datos. Suprime filas de datos de la base de datos. Modifica datos existentes en la base de datos.
DDL	**Definición de datos** CREATE TABLE DROP TABLE ALTER TABLE CREATE VIEW DROP VIEW CREATE INDEX DROP INDEX CREATE SYNOYM DROP SYNONYM	Añade una nueva tabla a la base de datos. Suprime una tabla de la base de datos. Modifica la estructura de una tabla existente. Añade una nueva vista a la base de datos. Suprime una vista de la base de datos. Construye un índice para una columna. Suprime el índice para una columna. Define un alias para un nombre de tabla. Suprime un alias para un nombre de tabla.
DCL	**Control de acceso** GRANT REVOKE **Control de transacciones** COMMIT ROLLBACK	Concede privilegios de acceso a usuarios. Suprime privilegios de acceso a usuarios Finaliza la transacción actual. Aborata la transacción actual.
PLSQL	**SQL Programático** DECLARE OPEN FETCH CLOSE	Define un cursor para una consulta. Abre un cursor para recuperar resultados de consulta. Recupera una fila de resultados de consulta. Cierra un cursor.

Componentes sintácticos

La mayoría de sentencias SQL tienen la misma estructura.

Todas comienzan por un verbo (select, insert, update, create), a continuación le sigue una o más cláusulas que nos dicen los datos con los que vamos a operar (from, where), algunas de estas son opcionales y otras obligatorias como es el caso del from.



4.5.5 Creación de tablas

En general, la mayoría de las bases de datos poseen potentes editores de bases que permiten la creación rápida y sencilla de cualquier tipo de tabla con cualquier tipo de formato.

Sin embargo, una vez la base de datos está alojada en el servidor, puede darse el caso de que queramos introducir una nueva tabla ya sea con carácter temporal (para gestionar un carrito de compra por ejemplo) o bien permanente por necesidades concretas de nuestra aplicación.

En estos casos, podemos, a partir de una sentencia SQL, crear la tabla con el formato que deseemos lo cual nos puede ahorrar más de un quebradero de cabeza.

Este tipo de sentencias son especialmente útiles para bases de datos como Mysql, las cuales trabajan directamente con comandos SQL y no por medio de editores.

Para crear una tabla debemos especificar diversos datos: El nombre que le queremos asignar, los nombres de los campos y sus características. Además, puede ser necesario especificar cuáles de estos campos van a ser índices y de qué tipo van a serlo.

La sintaxis de creación puede variar ligeramente de una base de datos a otra ya que los tipos de campo aceptados no están completamente estandarizados.

A continuación os explicamos someramente la sintaxis de esta sentencia y os proponemos una serie de ejemplos prácticos:

Sintaxis

```
Create Table nombre_tabla
(
nombre_campo_1 tipo_1
nombre_campo_2 tipo_2
nombre_campo_n tipo_n
Key(campo_x,...)
)
```

Pongamos ahora como ejemplo la creación de la tabla pedidos que hemos empleado en capítulos previos:

```
Create Table pedidos
(
id_pedido INT(4) NOT NULL AUTO_INCREMENT,
id_cliente INT(4) NOT NULL,
id_articulo INT(4)NOT NULL,
fecha DATE,
cantidad INT(4),
total INT(4), KEY(id_pedido,id_cliente,id_articulo)
)
```

En este caso creamos los campos *id* los cuales son considerados de tipo entero de una longitud especificada por el número entre paréntesis. Para *id_pedido* requerimos que dicho campo se incremente automáticamente (AUTO_INCREMENT) de una unidad a cada introducción de un nuevo registro para, de esta forma, automatizar su creación. Por otra parte, para evitar un mensaje de error, es necesario requerir que los campos que van a ser definidos como índices no puedan ser nulos (NOT NULL).

El campo *fecha* es almacenado con formato de fecha (DATE) para permitir su correcta explotación a partir de las funciones previstas a tal efecto.

Finalmente, definimos los índices enumerándolos entre paréntesis precedidos de la palabra KEY o INDEX.

Del mismo modo podríamos crear la tabla de *artículos* con una sentencia como ésta:

```
Create Table articulos
(
id_articulo INT(4) NOT NULL AUTO_INCREMENT,
titulo VARCHAR(50),
```

```
autor VARCHAR(25),
editorial VARCHAR(25),
precio REAL,
KEY(id_articulo)
)
```

En este caso puede verse que los campos alfanuméricos son introducidos de la misma forma que los numéricos. Volvemos a recordar que en tablas que tienen campos comunes es de vital importancia definir estos campos de la misma forma para el buen funcionamiento de la base.

Muchas son las opciones que se ofrecen al generar tablas. No vamos a tratarlas detalladamente pues sale de lo estrictamente práctico. Tan sólo mostraremos algunos de los tipos de campos que pueden ser empleados en la creación de tablas con sus características:

Tipo	Bytes	Descripción
INT o INTEGER	4	Números enteros. Existen otros tipos de mayor o menor longitud específicos de cada base de datos.
DOUBLE o REAL	8	Números reales (grandes y con decimales). Permiten almacenar todo tipo de número no entero.
CHAR	1/caracter	Alfanuméricos de longitud fija predefinida
VARCHAR	1/caracter+1	Alfanuméricos de longitud variable
DATE	3	Fechas, existen multiples formatos específicos de cada base de datos
BLOB	1/caracter+2	Grandes textos no indexables
BIT o BOOLEAN	1	Almacenan un bit de información (verdadero o falso)

4.5.6 Insertar un nuevo registro

Los registros pueden ser introducidos a partir de sentencias que emplean la instrucción Insert.

La sintaxis utilizada es la siguiente:

Insert Into nombre_tabla (nombre_campo1, nombre_campo2,...) Values (valor_campo1, valor_campo2...)

Un ejemplo sencillo a partir de nuestra tabla modelo es la introducción de un nuevo cliente lo cual se haría con una instrucción de este tipo:

Insert Into clientes (nombre, apellidos, direccion, poblacion, codigopostal, email, pedidos) Values ('Perico', 'Palotes', 'Percebe n°13', 'Lepe', '123456', 'perico@desarrolloweb.com', 33)

Como puede verse, los campos no numéricos o booleanos van delimitados por apostrofes: '. También resulta interesante ver que el código postal lo hemos guardado como un campo no numérico. Esto es debido a que en determinados paises (Inglaterra, como no) los codigos postales contienen también letras.

Por supuesto, no es imprescindible rellenar todos los campos del registro. Eso sí, puede ser que determinados campos sean necesarios. Estos campos necesarios pueden ser definidos cuando construimos nuestra tabla mediante la base de datos.

4.5.7 Borrar un registro

Para borrar un registro nos servimos de la instrucción Delete. En este caso debemos especificar cual o cuales son los registros que queremos borrar. Es por ello necesario establecer una selección que se llevara a cabo mediante la cláusula Where.

La forma de seleccionar se verá detalladamente en capítulos posteriores. Por ahora nos contentaremos de mostrar cuál es el tipo de sintaxis utilizado para efectuar estas supresiones:

Delete From nombre_tabla Where condiciones_de_selección

Si queremos por ejemplo borrar todos los registros de los clientes que se llamen Perico lo haríamos del siguiente modo:

Delete From clientes Where nombre='Perico'

Hay que tener cuidado con esta instrucción ya que si no especificamos una condición con Where, lo que estamos haciendo es **borrar toda la tabla**:

4.5.8 Actualizar un registro

Update es la instrucción que nos sirve para modificar nuestros registros. Como para el caso de Delete, necesitamos especificar por medio de Where cuáles son los registros en los que queremos hacer efectivas nuestras modificaciones. Además, obviamente, tendremos que especificar cuáles son los nuevos valores de los campos que deseamos actualizar. La sintaxis es de este tipo:

Update nombre_tabla Set nombre_campo1 = valor_campo1, nombre_campo2 = valor_campo2,... Where condiciones_de_selección

Un ejemplo aplicado:

Update clientes Set nombre='José' Where nombre='Pepe'

Mediante esta sentencia cambiamos el nombre Pepe por el de José en todos los registros cuyo nombre sea Pepe.

Aquí también hay que ser cuidadoso de no olvidarse de usar Where, de lo contrario, modificaríamos todos los registros de nuestra tabla.

5 Especificació

En este apartado se presentaran los diagramas de casos de uso de cada uno de los diferentes roles de usuarios de la aplicación, también se hará una explicación de cada uno de estos de acuerdo con los requerimientos funcionales que se han descrito anteriormente. Los casos de uso se encargan de definirlos escenarios en los que se realizan las acciones por parte del usuario de la aplicación, definiendo los elementos que intervienen y representan los acontecimientos y las diferentes vías de ejecución que pueden existir para cada uno de los casos.

Antes de esquematizar y definir los casos de uso describiré la jerarquía que existe dentro de la aplicación. Se pueden definir 2 tipos de usuarios dentro de la aplicación, cada uno tiene distintos privilegios y se pueden ordenar verticalmente según su especificación. Se pueden observar 2 niveles en la aplicación, cada nivel es superior al inferior y puede realizar todas las acciones que el nivel inferior más algunas otras que sólo puede ejecutar ese mismo nivel.

El esquema de actores quedaría así:

Esta estructura ha sido elegida puesto que inicialmente los usuarios básicos de un CMS son el publicador y el administrador, aunque durante el transcurso de la implementación y desarrollo del proyecto he ido pensando en posibles roles que podrían ser añadidos a la aplicación en futuras actualizaciones, como por ejemplo la opción de tener un rol para usuarios web sin acceso al CMS pero sí a una parte privada de la web, o por ejemplo el rol becario que es el que introduciría el contenido, pero sin poder llegar a publicarlo, puesto que esta acción seria del rol publicador.

Administrador

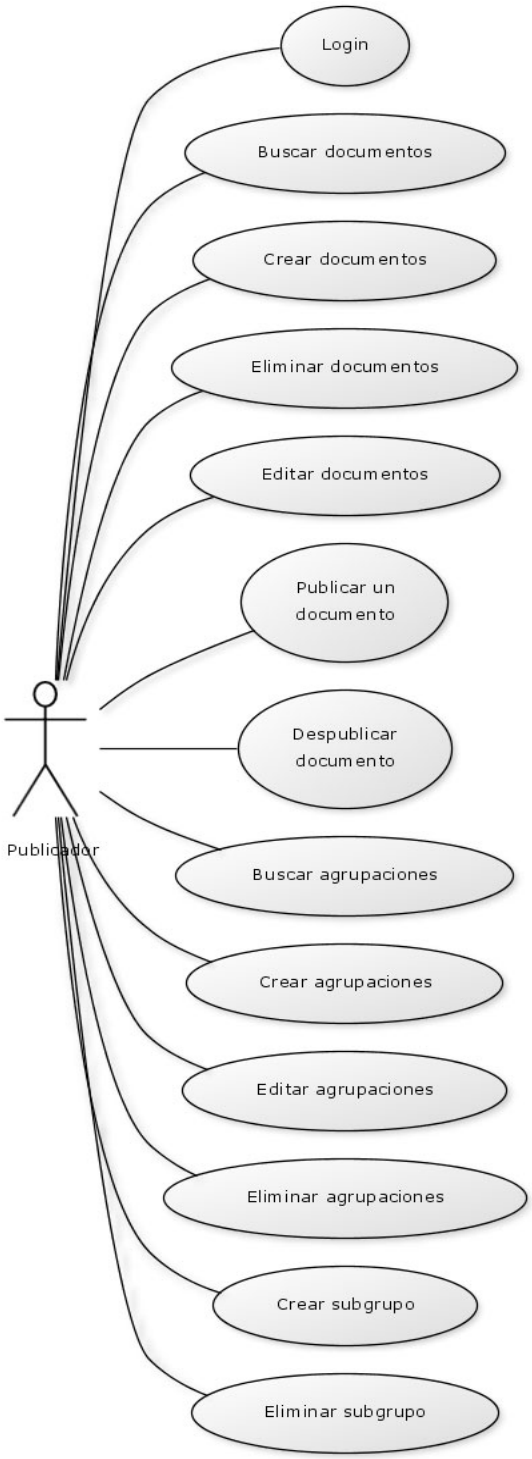
Este usuario que tiene como rol administrador está definido y pensado para que no tenga ninguna restricción o lo que es lo mismo que disponga de todos los privilegios de la aplicación.

Publicador

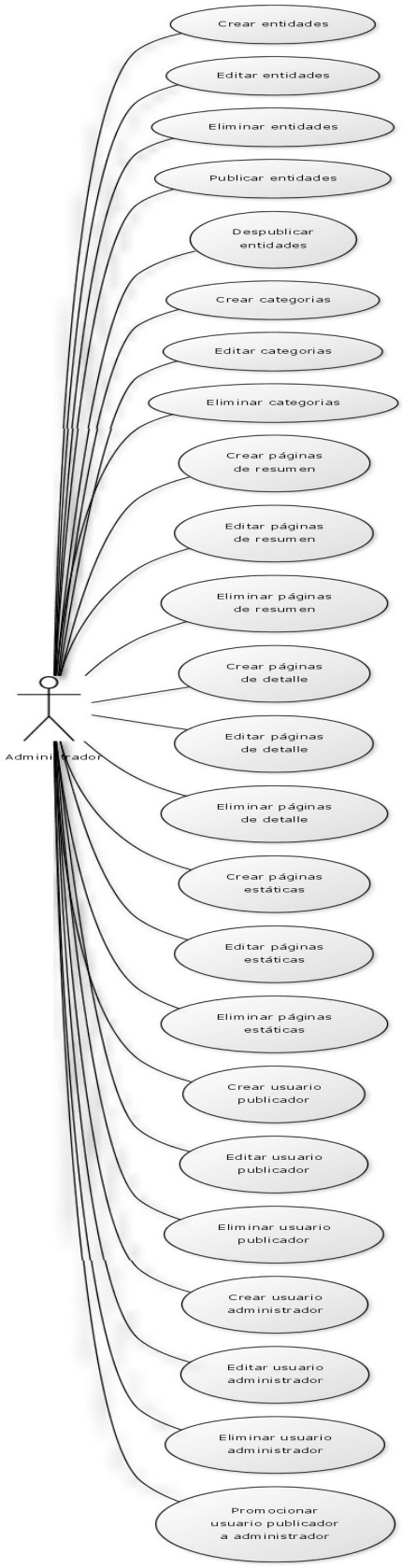
Los usuarios que dispongan del rol publicador el único privilegio del que dispondrán será el de administrar contenido (crear, editar, eliminar, publicar/despublicar) y también podrán administrar las agrupaciones (crear, editar y eliminar).

5.1 Diagrama de casos de uso

5.1.1 Casos de uso del rol publicador



5.1.2 Casos de uso del rol administrador



5.2 Descripción de casos de uso

En el siguiente apartado procedo a describir los casos de uso que se han especificado ofreciendo información relacionada con el intercambio de información que se produce entre el usuario y la aplicación, describiendo qué tipos de cursos posibles puede seguir cada caso de uso y las interacciones que se producen y como se llegan a ser ejecutados en su totalidad.

5.2.1 Caso de uso de login

La aplicación tiene un mecanismo de acceso y de autentificación para controlar a que rol pertenece cada usuario y para consolar el acceso a la aplicación. Para acceder a la aplicación se tiene que proceder a realizar este caso de uso.

Login

Actor	Publicador	
Entrada	Nombre del usuario y contraseña	
Salida	-	
Resumen	El usuario accede a la aplicación	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario introduce el domino donde se encuentra la aplicación en un navegador		
		2.- El sistema redirige y muestra la pantalla de login
3.- El usuario introduce su nombre de usuario y la contraseña		
		4.- El sistema comprueba los datos y configura la vista de opciones de acuerdo al rol del usuario
5.- El usuario ha accedido a la aplicación		

Curso alternativo	
	4.- Los datos del usuario o contraseña no son válidos y se envía un mensaje de error
5.- Aparece un mensaje de error en la misma pantalla de login	

5.2.2 Caso de uso de buscar documentos

En la sección de contenido encontramos un árbol donde encontramos la pestaña de documentos y categorías. Si hacemos clic en la opción de documentos se nos abrirá un frame para poder buscar un documento con la ayuda de filtros de búsqueda.

Buscard documentos

Actor	Publicador	
Entrada	-	
Salida	Resumen de documentos	
Resumen	Nos aparecerá un resumen de documentos	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario rellena los campos del filtro del buscador y hace clic en el botón buscar		
		2.- El sistema realiza un búsqueda con los criterios de búsqueda que el usuario ha marcado en el filtro.
		3.- Muestra todos los registros formando un resumen de documentos
4.- El usuario visualiza todos los documentos que cumplen el filtro.		

5.2.3 Caso de uso de crear documentos

Una vez ubicados dentro de los contenido y en la entidad en la que queremos agregar un documento haremos clic en el botón nuevo, una vez realizada esta acción nos aparecerá una nueva ventana con campos que deberemos rellenar para hacer esta nueva inserción.

Crear documentos

Actor	Publicador
Entrada	Información relevante al documento
Salida	Agregaremos un nuevo documento al sistema
Resumen	Crearemos un documento de la entidad deseada
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario hará clic en el botón nuevo una vez haya seleccionado la entidad a la que pertenecerá	
	2.- El sistema abrirá una ventana con los campos que tiene la entidad
3.- El usuario rellenado los campos del documento y hace clic en el botón guardar	
	4.- El sistema agrega este nuevo documento en la base de datos
	5.- El sistema vuelve a mostrar el resumen de documentos

5.2.4 Caso de uso de eliminar un documento

El usuario localizará el elemento que quiere eliminar y hará clic en la X de color rojo que se encuentra a la derecha del registro.

Eliminar un documento

Actor	Publicador	
Entrada	-	
Salida	Elimina el documento seleccionado	
Resumen	El usuario eliminará un documento	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localizará el documento que quiere eliminar y hará clic en la X		
		2- El sistema elimina el registro de la base de datos y del greed de resumen de documentos

5.2.5 Caso de uso de editar un documento

Una vez el usuario ha localizado el documento que quiere editar hace clic sobre el registro o sobre el lápiz de edición y se abre el formulario de la entidad para que modifique la información necesaria, tras estas modificaciones hace clic en el botón guardar.

Editar un documento

Actor	Publicador	
Entrada	Información referente al documento	
Salida	-	
Resumen	El usuario realiza una modificación en un documento	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localiza el documento que desea editar y hace clic en el registro		

	2.- El sistema carga el formulario de la entidad con la información del documento
3.- El usuario edita la información necesaria y hace clic en botón guardar	
	4.- El sistema actualiza el documento y muestra de nuevo el resumen de documentos de la entidad

5.2.6 Caso de uso de publicar un documento

El usuario localiza el documento que quiere publicar y hace clic sobre el registro del documento, cuando el sistema haya cargado la información referente al documento que estamos tratando, hace clic sobre el botón publicar.

Publicar un documento

Actor	Publicador	
Entrada	Información referente al documento	
Salida	-	
Resumen	El usuario publica un documento	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localiza el documento que desea editar y hace clic en el registro		
		2.- El sistema carga el formulario de la entidad con la información del documento
3.- El usuario hace clic sobre el botón publicar		
		4.- El sistema actualiza el documento y muestra de nuevo el resumen de documentos de la entidad

5.2.7 Caso de uso de Despublicar un documento

El usuario localiza el documento que quiere publicar y hace clic sobre el registro del documento, cuando el sistema haya cargado la información referente al documento que estamos tratando, hace clic sobre el botón despublicar.

Despublicar un documento

Actor	Publicador	
Entrada	Información referente al documento	
Salida	-	
Resumen	El usuario despublica un documento	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localiza el documento que desea editar y hace clic en el registro		
		2.- El sistema carga el formulario de la entidad con la información del documento
3.- El usuario hace clic sobre el botón despublicar		
		4.- El sistema actualiza el documento y muestra de nuevo el resumen de documentos de la entidad

5.2.8 Caso de uso buscar agrupaciones

Para realizar esta acción primeramente nos tenemos que situar en la sección de agrupaciones escribir el nombre del grupo y hacer clic en el botón buscar, el sistema nos filtrara las agrupaciones que tengamos para encontrar la que deseamos.

Buscar agrupaciones

Actor	Publicador	
Entrada	-	
Salida	Resumen de agrupaciones	
Resumen	Nos aparecerá un resumen de agrupaciones	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario rellena el campo de texto y hace clic en el botón buscar		
		2.- El sistema realiza un búsqueda con los criterios de búsqueda que el usuario ha marcado en el filtro.
		3.- Muestra todos los registros formando un resumen de documentos
4.- El usuario visualiza todos los documentos que cumplen el filtro.		

5.2.9 Caso de uso de crear una agrupación

Una vez ubicados dentro de la sección de agrupaciones hacemos clic en el botón nuevo y nos cargara el formulario para crear un grupo en el que tendremos que rellenar los campos y decir a qué entidades afectará esta agrupación, para finalizar esta acción haremos clic en botón guardar.

Crear una agrupación

Actor	Publicador	
Entrada	Información referente a la agrupación	
Salida	-	
Resumen	El usuario creará una agrupación	
Curso típico de acontecimientos		
Actor		Sistema
1.-El usuario hace clic en el botón nuevo de la sección de agrupaciones		
		2.- El sistema carga el formulario para crear un

	nuevo grupo
3.- El usuario rellena los campos pertinentes y agrega las entidades a las que afectara esta nueva agrupación y hace clic en el botón guardar	
	4.- El sistema agrega esta nueva agrupación a la base de datos y relaciona la entidades correspondientes con esta
	5.- El sistema enseña el greed de resumen de las agrupaciones del cms

5.2.10 Caso de uso de editar una agrupación

Para realizar esta acción es necesario ubicarnos dentro de la sección de agrupaciones y localizar la que queremos editar y hacer clic encima del registro para que el sistema abra el formulario de esta agrupación y de esta manera poder editarlo y poder guardar los cambios realizados.

Editar una agrupación

Actor	Publicador
Entrada	Información referente a la agrupación
Salida	-
Resumen	El usuario realiza una modificación en una agrupación
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localiza el grupo que desea editar y hace clic en el registro	
	2.- El sistema carga el formulario del grupo con la información de este
3.- El usuario edita la información necesaria y hace clic en botón guardar	

	4.- El sistema actualiza laagrupación y muestra de nuevo el resumen de grupos

5.2.11 Caso de uso de eliminar una agrupación

Para realizar esta acción será suficiente con que el usuario encuentre dentro de la sección de agrupaciones el grupo que desea eliminar y haga clic en la X roja que se encuentra a la derecha del registro.

Eliminar una agrupación

Actor	Publicador	
Entrada	-	
Salida	Elimina el grupo seleccionado	
Resumen	El usuario eliminará un grupo	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localizará el grupo que quiere eliminar y hará clic en la X del registro		
		2- El sistema elimina el registro de la base de datos eliminado también los vínculos con los documentos y muestra el greed de resumen de agrupaciones

5.2.12 Caso de uso de crear subgrupo

Para crear un subgrupo tendremos que localizar el grupo del que colgará este subgrupo que pretendemos crear y hacer clic sobre el, una vez realizada esta acción abajo a la derecha encontraremos un botón nuevo que tendremos que presionar para que el sistema nos cargue el formulario para crear este tipo de elementos, una vez rellenados los campos de este formulario debemos guardar este subgrupo.

Crear un subgrupo

Actor	Publicador	
Entrada	Información referente al subgrupo	
Salida	-	
Resumen	El usuario creara un subgrupo	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localiza el grupo al que pretende agregarle un subgrupo y hace clic sobre el registro		
		2.- El sistema carga el formulario con la información del grupo seleccionado
3.- El usuario hace clic en el botón nuevo		
		4.- El sistema carga en la parte inferior el formulario de creación del subgrupo
5.- El usuario rellena los campos del formulario y presiona el botón guardar		
		6.- El sistema muestra el formulario del grupo y en la parte inferior un rectángulo con los subgrupos que este tiene

5.2.13 Caso de uso de eliminar un subgrupo

Para realizar esta acción el usuario debe localizar dentro de la sección de agrupaciones el grupo del que cuelga el subgrupo que desea eliminar una vez localizado debe hacer clic encima del registro del mismo, el sistema abrirá el formulario de registro y en la parte inferior verá una ventana con todos los subgrupos que tiene este grupo, deberá seleccionar el que desee y hacer clic en el botón borrar.

Eliminar un subgrupo

Actor	Publicador
Entrada	-
Salida	-
Resumen	El usuario eliminara un subgrupo
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localiza el grupo del que pretende eliminar uno de sus subgrupos	
	2.- El sistema carga el formulario de registro del grupo junto con sus subgrupos
3.- El usuario selecciona el subgrupo que quiere eliminar y hace clic en el botón borrar	
	4.- El sistema elimina el subgrupo de la lista y desasigna todos los documentos que tenia asignados

5.2.14 Caso de uso de crear entidades

Para crear una entidad debemos situarnos en la sección de administración y hacer clic en entidades, una vez realizada esta acción debemos hacer clic en el botón nuevo y el sistema nos cargará una pagina donde deberemos configurar los parámetros de la entidad.

Crear entidades

Actor	Administrador
Entrada	Información relevante a la entidad
Salida	Agregaremos una nueva entidad al sistema
Resumen	Crearemos una entidad

Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario hará clic en el botón nuevo de la sección de entidades	
	2.- El sistema abrirá una ventana con el formulario de creación de una entidad
3.- El usuario configura la entidad y añade los campos que deas que tenga esta entidad y hace clic en el botón guardar	
	4.- El sistema agrega esta nueva entidad a la base de datos
	5.- El sistema vuelve a mostrar el resumen de entidades

5.2.14 Caso de uso de editar una entidad

Para que el usuario pueda editar una entidad debe localizarla dentro del greed de resumen de estas y hacer clic sobre el registro de la misma para que el sistema abra el formulario de la entidad y finalmente que el usuario pueda modificar la información necesaria para poder guardar los cambios.

Editar una entidad

Actor	Administrador
Entrada	Información referente a la entidad
Salida	-
Resumen	El usuario realiza una modificación en una entidad
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localiza la entidad que desea editar y hace clic en el registro	

	2.- El sistema carga el formulario de la entidad con la información
3.- El usuario edita la información necesaria y hace clic en botón guardar	
	4.- El sistema actualiza la nueva entidad y muestra de nuevo el resumen de entidades

5.2.15 Caso de uso de eliminar una entidad

Para realizar esta acción debemos localizar la entidad que queremos eliminar dentro del greed de resumen de estas y hacer clic en el botón marcado con una X roja situado a la izquierda del registro.

Eliminar una entidad

Actor	Administrador	
Entrada	-	
Salida	Elimina la entidad seleccionado	
Resumen	El usuario eliminará una entidad	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localizará la entidad que quiere eliminar y hará clic en la X		
		2- El sistema elimina el registro de la base de datos y del greed de resumen de entidades

5.2.16 Caso de uso de publicar una entidad

El usuario localiza la entidad que quiere publicar y hace clic sobre el registro de la entidad, cuando el sistema haya cargado la información referente a la entidad que estamos tratando, hace clic sobre el botón publicar.

Publicar una entidad

Actor	Administrador	
Entrada	Información referente a la entidad	
Salida	-	
Resumen	El usuario publica una entidad	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localiza laentidad que desea editar y hace clic en el registro		
		2.- El sistema carga el formulario de la entidad con la información
3.- El usuario hace clic sobre el botón publicar		
		4.- El sistema actualiza laentidad y muestra de nuevo el resumenentidades

5.2.17 Caso de uso de Despublicar una entidad

El usuario localiza la entidad que quiere publicar y hace clic sobre el registro de la entidad, cuando el sistema haya cargado la información referente a la entidad que estamos tratando, hace clic sobre el botón despublicar.

Despublicar unaentidad

Actor	Administrador	
Entrada	Información referente a la entidad	
Salida	-	
Resumen	El usuario publica una entidad	
Curso típico de acontecimientos		
Actor		Sistema

1.- El usuario localiza la entidad que desea editar y hace clic en el registro	
	2.- El sistema carga el formulario de la entidad con la información
3.- El usuario hace clic sobre el botón despublicar	
	4.- El sistema actualiza laentidad y muestra de nuevo el resumenentidades

5.2.18 Caso de uso de crear una categoria

Una vez ubicados dentro de la sección de categorías hacemos clic en el botón nuevo y nos cargará el formulario para crear una categoría en el que tendremos que rellenar los campos y hacer clic en botón guardar.

Crear una categoria

Actor	Administrador
Entrada	Información referente a la categoría
Salida	-
Resumen	El usuario creará una agrupación
Curso típico de acontecimientos	
Actor	Sistema
1.-El usuario hace clic en el botón nuevo de la sección de categorías	
	2.- El sistema carga el formulario para crear una nuevacategoría
3.- El usuario rellena los campos pertinentes y hace clic en el botón guardar	

	4.- El sistema agrega esta nueva categoría a la base de datos
	5.- El sistema enseña el greed de resumen de las categorías del cms

5.2.19 Caso de uso de editar una categoría

Para realizar esta acción es necesario ubicarnos dentro de la sección de categorías y localizar la que queremos editar y hacer clic encima del registro para que el sistema abra el formulario de esta categoría y de esta manera poder editarla y poder guardar los cambios realizados.

Editar una categoría

Actor	Administrador	
Entrada	Información referente a la categoría	
Salida	-	
Resumen	El usuario realiza una modificación en una categoría	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localiza lacategoría que desea editar y hace clic en el registro		
		2.- El sistema carga el formulario de la categoría con la información de esta
3.- El usuario edita la información necesaria y hace clic en botón guardar		
		4.- El sistema actualiza lacategoría y muestra de nuevo el resumen de grupos

5.2.20 Caso de uso de eliminar una categoría

Para realizar esta acción será suficiente con que el usuario encuentre dentro de la sección de categorías la categoría que desea eliminar y haga clic en la X roja que se encuentra a la derecha del registro.

Eliminar una categoría

Actor	Administrador	
Entrada	-	
Salida	Elimina la categoría seleccionado	
Resumen	El usuario eliminará una categoría	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localizará lacategoría que quiere eliminar y hará clic en la X del registro		
		2- El sistema elimina el registro de la base de datos eliminado también los vínculos con las entidadesy muestra el greed de resumen de categorías

5.2.21 Caso de uso de crea una página de resumen

Para crear una página de resumen debemos situarnos en la sección de páginas > resumen y hacer clic en el botón nuevo y el sistema nos cargará una página donde deberemos configurar la página que queremos generar. Finalmente debemos hacer clic en el botón guardar.

Crear una página de resumen

Actor	Administrador	
Entrada	Código HTML	
Salida	Agregaremos una nueva página de resumen	
Resumen	Crearemos una página de resumen	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario hará clic en el botón nuevo de la sección de paginas > resumen		

	2.- El sistema abrirá una ventana con el formulario de creación de páginas
3.- El usuario rellenará los campos y las pestañas necesarias para la página y hará clic en el botón guardar	
	4.- El sistema agrega esta nueva página a la base de datos
	5.- El sistema vuelve a mostrar el resumen de páginas

5.2.22 Caso de uso de editar una página de resumen

Para que un usuario pueda editar una página de resumen debe localizarla dentro del greed de resumen de éstas y hacer clic sobre el registro de la misma para que el sistema abra el formulario de la página y finalmente que el usuario pueda modificar la información necesaria para poder guardar los cambios.

Editar una página de resumen

Actor	Administrador
Entrada	Código HTML
Salida	-
Resumen	El usuario realiza una modificación en una página de resumen
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localiza la página que desea editar y hace clic en el registro	
	2.- El sistema carga el formulario de la página con la información
3.- El usuario edita la información necesaria y	

hace clic en botón guardar	
	4.- El sistema actualiza la nueva página y muestra de nuevo el resumen de páginas

5.2.23 Caso de uso de eliminar una página de resumen

Para realizar esta acción debemos localizar la página de resumen que queremos eliminar dentro del greed de resumen de estas y hacer clic en el botón marcado con una X roja situado a la izquierda del registro.

Eliminar una página de resumen

Actor	Administrador
Entrada	-
Salida	Elimina la página seleccionado
Resumen	El usuario eliminará una página

Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localizará lapágina que quiere eliminar y hará clic en la X	
	2- El sistema elimina el registro de la base de datos y mostrara el greed de resumen de páginas

5.2.24 Caso de uso de crear una página de detalle

Para crear una página de detalle debemos situarnos en la sección de paginas > resumen y hacer clic en el botón nuevo y el sistema nos cargará una pagina donde deberemos configurar la página que queremos generar finalmente debemos hacer clic en el botón guardar.

Crear una página de detalle

Actor	Administrador
Entrada	Código HTML
Salida	Agregaremos una nueva página de detalle
Resumen	Crearemos una página de detalle
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario hará clic en el botón nuevo de la sección de páginas > detalle	
	2.- El sistema abrirá una ventana con el formulario de creación de páginas
3.- El usuario rellenará los campos y las pestañas necesarias para la página y hará clic en el botón guardar	
	4.- El sistema agrega esta nueva página a la base de datos
	5.- El sistema vuelve a mostrar el detalle de páginas

5.2.25 Caso de uso de editar una página de detalle

Para que un usuario pueda editar una página de detalle debe localizarla dentro del greed de resumen de estas y hacer clic sobre el registro de la misma para que el sistema abra el formulario de la página y finalmente que el usuario pueda modificar la información necesaria para poder guardar los cambios.

Editar una página de resumen

Actor	Administrador
Entrada	Código HTML
Salida	-
Resumen	El usuario realiza una modificación en una página de detalle

Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localiza la página que desea editar y hace clic en el registro	
	2.- El sistema carga el formulario de la página con la información
3.- El usuario edita la información necesaria y hace clic en botón guardar	
	4.- El sistema actualiza la nueva página y muestra de nuevo el resumen de páginas

5.2.26 Caso de uso de eliminar una página de detalle

Para realizar esta acción debemos localizar la página de detalle que queremos eliminar dentro del greed de resumen de éstas y hacer clic en el botón marcado con una X roja situado a la izquierda del registro.

Eliminar una página de resumen

Actor	Administrador
Entrada	-
Salida	-
Resumen	El usuario eliminará una página

Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localizará la página que quiere eliminar y hará clic en la X	
	2- El sistema elimina el registro de la base de datos y mostrará el greed de resumen de páginas

5.2.27 Caso de uso de crear una página de estática

Para crear una página de resumen debemos situarnos en la sección de páginas > estática y hacer clic en el botón nuevo y el sistema nos cargara una pagina donde deberemos configurar la página que queremos generar finalmente debemos hacer clic en el botón guardar.

Crear una página de estática

Actor	Administrador	
Entrada	Código HTML	
Salida	Agregaremos una nueva página de estática	
Resumen	Crearemos una página de estática	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario hará clic en el botón nuevo de la sección de páginas >estática		
		2.- El sistema abrirá una ventana con el formulario de creación de páginas
3.- El usuario rellenará los campos y las pestañas necesarias para la página y hará clic en el botón guardar		
		4.- El sistema agrega esta nueva página a la base de datos
		5.- El sistema vuelve a mostrar el resumen de páginas

5.2.28 Caso de uso de editar una página de estática

Para que un usuario pueda editar una página de estática debe localizarla dentro del greed de resumen de éstas y hacer clic sobre el registro de la misma para que el sistema abra el formulario de la página y finalmente que el usuario pueda modificar la información necesaria para poder guardar los cambios.

Editar una página de estática

Actor	Administrador	
Entrada	Código HTML	
Salida	-	
Resumen	El usuario realiza una modificación en una página de estática	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localiza la página que desea editar y hace clic en el registro		
		2.- El sistema carga el formulario de la página con la información
3.- El usuario edita la información necesaria y hace clic en botón guardar		
		4.- El sistema actualiza la nueva página y muestra de nuevo el resumen de páginas

5.2.29 Caso de uso de eliminar una página de estática

Para realizar esta acción debemos localizar la página de estática que queremos eliminar dentro del greed de resumen de éstas y hacer clic en el botón marcado con una X roja situado a la izquierda del registro.

Eliminar una página de estática

Actor	Administrador
Entrada	-
Salida	-
Resumen	El usuario eliminará unapágina
Curso típico de acontecimientos	

Actor	Sistema
1.- El usuario localizará la página que quiere eliminar y hará clic en la X	
	2- El sistema elimina el registro de la base de datos y mostrará el greed de resumen de páginas

5.2.30 Caso de uso de crear usuario publicador

Para crear un usuario con el rol publicador debemos situarnos dentro de la sección configuración > usuarios > publicador y hacer clic en el botón nuevo el sistema nos abrirá un formulario que el usuario deberá rellenar con su información y finalmente hacer clic en el botón guardar para que quede registrado en el sistema.

Crear un usuario publicador

Actor	Administrador
Entrada	Información relevante al usuario
Salida	Agregaremos un nuevo usuario
Resumen	Crearemos un usuario
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario hará clic en el botón nuevo de la sección de usuarios> publicador	
	2.- El sistema abrirá una ventana con el formulario de creación de usuarios
3.- El usuario rellenará los campos y las pestañas necesarias para el usuario y hará clic en el botón guardar	
	4.- El sistema agrega este nuevousuario a la base de datos
	5.- El sistema vuelve a mostrar el resumen de usuarios

5.2.31 Caso de uso de editar un usuario publicador

Para que un usuario pueda editar un usuario con el rol publicador primeramente deberá localizarlo dentro del greed de resumen pertenecientes a este rol y hacer clic encima del registro deseado, una vez realizada esta acción el sistema nos cargará el formulario con la información perteneciente al usuario y solo faltará editar la que creamos conveniente y hacer clic en el botón guardar.

Editar un usuario publicador

Actor	Administrador
Entrada	Información referente al usuario
Salida	-
Resumen	El usuario realiza una modificación en un usuario publicador
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localiza el usuario que desea editar y hace clic en el registro	
	2.- El sistema carga el formulario del usuario con la información
3.- El usuario edita la información necesaria y hace clic en botón guardar	
	4.- El sistema actualiza el usuario y muestra de nuevo el resumen de estos

5.2.32 Caso de uso de eliminar un usuario publicador

Para realizar esta acción debemos localizar el usuario con el rol publicador que queremos eliminar dentro del greed de resumen de estos y hacer clic en el botón marcado con una X roja situado a la izquierda del registro.

Eliminar un usuario publicador

Actor	Administrador
Entrada	-
Salida	Elimina un usuario
Resumen	El usuario eliminará un usuario publicador
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localizará el usuario que quiere eliminar y hará clic en la X	
	2- El sistema elimina el registro de la base de datos y mostrara el greed de resumen de usuarios

5.2.33 Caso de uso de crear usuario administrador

Para crear un usuario con el rol publicador debemos situarnos dentro de la sección configuración > usuarios > publicador y hacer clic en el botón nuevo el sistema nos abrirá un formulario que el usuario deberá rellenar con su información y finalmente hacer clic en el botón guardar para que quede registrado en el sistema.

Crear un usuario administrador

Actor	Administrador
Entrada	Información relevante al usuario
Salida	Agregaremos un nuevo usuario
Resumen	Crearemos un usuario administrador
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario hará clic en el botón nuevo de la sección de usuarios > administrador	
	2.- El sistema abrirá una ventana con el formulario de creación de usuarios
3.- El usuario rellenará los campos y las pestañas necesarias para el usuario y hará clic en el botón guardar	

	4.- El sistema agrega este nuevo usuario a la base de datos
	5.- El sistema vuelve a mostrar el resumen de usuarios

5.2.34 Caso de uso de editar un usuario administrador

Para que un usuario pueda editar un usuario con el rol publicador primeramente deberá localizarlo dentro del greed de resumen pertenecientes a este rol y hacer clic encima del registro deseado, una vez realizada esta acción el sistema nos cargará el formulario con la información perteneciente al usuario y solo faltará editar la que creamos conveniente y hacer clic en el botón guardar.

Editar un usuario administrador

Actor	Administrador
Entrada	Información referente al usuario
Salida	-
Resumen	El usuario realiza una modificación en un usuario administrador
Curso típico de acontecimientos	
Actor	Sistema
1.- El usuario localiza el usuario que desea editar y hace clic en el registro	
	2.- El sistema carga el formulario del usuario con la información
3.- El usuario edita la información necesaria y hace clic en botón guardar	
	4.- El sistema actualiza el usuario y muestra de nuevo el resumen de estos

5.2.35 Caso de uso de eliminar un usuario administrador

Para realizar esta acción debemos localizar el usuario con el rol administrador que queremos eliminar dentro del greed de resumen de estos y hacer clic en el botón marcado con una X roja situado a la izquierda del registro.

Eliminar un usuario administrador

Actor	Administrador	
Entrada	-	
Salida	Elimina un usuario	
Resumen	El usuario eliminará un usuario administrador	
Curso típico de acontecimientos		
Actor		Sistema
1.- El usuario localizará el usuario que quiere eliminar y hará clic en la X		
		2- El sistema elimina el registro de la base de datos y mostrará el greed de resumen de usuarios

6 Peticiones al CMS

En este apartado se explicara como realizar peticiones al CMS, para poder mostrar las plantillas con los documentos que hemos creado.

Para realizar peticiones al CMS deberemos llamar a la siguiente url www.tudominio.com/admin/web/manage.php? a continuación del interrogante será donde debamos indicar los parámetros para filtrar el contenido, los diferentes parámetros que podemos usar son:

Tipo: El parámetro tipo será el que nos sirva para decirle que tipo de plantilla va a mostrar la aplicación. Hay diferentes valores que le podamos dar a la variable tipo: 2 para plantillas estáticas, 3 para plantilla de detalle, 5 para plantilla de resumen, 9 para resumen de hijos.

Plantilla: A esta variable le tendremos que pasar la id de la plantilla que tenga la que queramos mostrar.

Entidad: A la variable entidad le pasaremos la id de la entidad que vayamos a tratar.

ParentId: Le pasaremos la id del padre del que vayamos a sacar sus hijos.

SearchText: A esta variable le pasaremos una cadena de texto que queramos buscar.

Typerel: Este variable recibirá valor 1 o 2 en función de si la relación es padres/hijos o documentos relacionados

Id: Este valor recibirá la id del documento que tenemos que mostrar.

En futuras mejoras del CMS me gustaría poder añadir mas tags para futuras funcionalidades que pueda requerir algún portal web.

A continuación pondré un ejemplo de cómo serían algunas posibles peticiones:

www.tudominio.com/admin/web/manage.php?tipo=5&entidad=2&plantilla=7"

Este ejemplo sacaría un resumen con la plantilla con id 7 y mostrará todos los documentos que pertenezcan a la entidad con id numero 2.

www.tudominio.com/admin/web/manage.php?entidad=13&tipo=3&plantilla=28&Id=23

Este ejemplo nos sacará un detalle usando la plantilla 28 en el que nos mostrará el documento con id número 23 que pertenece a la entidad numero 13.

Usando esta combinación de variables será la manera que tendremos para poder sacará la información del CMS.

7 Manual de usuario

En primer lugar explicaremos que una vez instalado y configurado nuestro CMS podemos acceder a él de la siguiente manera www.nombredeldomino.com/admin

Es importante diferenciar que en un lado tenemos nuestra web www.nombredeldominio.com y colgando de ésta tenemos otra web dentro de la carpeta admin que se encuentra en la raíz de nuestro servidor que se llama admin.

7.1 Login

Una vez hemos accedido a esta página nos encontraremos un formulario donde tendremos que validarnos como usuarios registrados en el CMS.

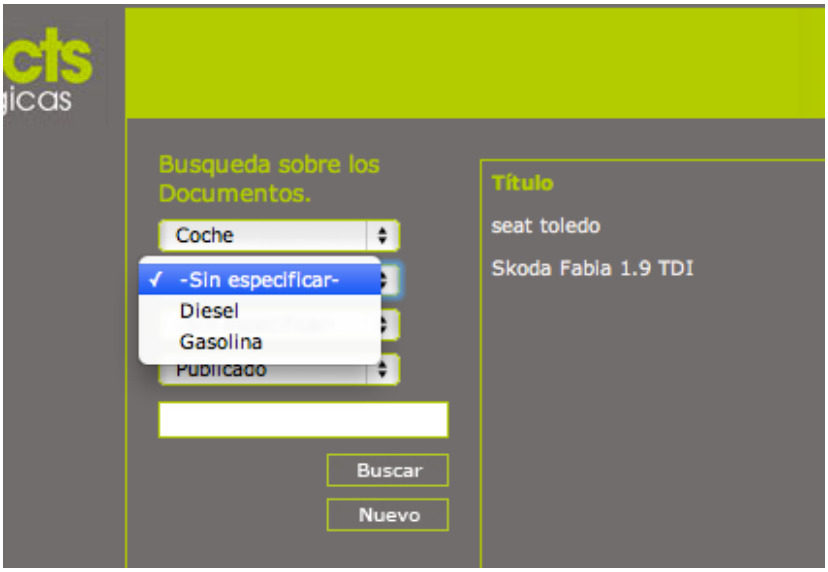
7.2 Contenido

Una vez hemos accedido al CMS será donde podremos apreciar que está formado por tres grandes bloques: Contenidos, Administración y Configuración es importante saber que la visualización de cada uno de estos botones va vinculado a los permisos del usuario.

Cada uno de estos botones corresponde con un apartado diferente de la aplicación el primero de estos es Contenido que será el que se abre por defecto siempre al iniciar con cualquier usuario. En esta sección será donde el usuario final o en su defecto el administrador podrá mantener actualizada la información que quiera reflejar en el sitio web, en esta sección se pueden crear, editar y eliminar contenido. Con el fin de poder encontrar el contenido rápidamente esta sección dispone de un buscador que nos facilitará la labor de buscar la información.

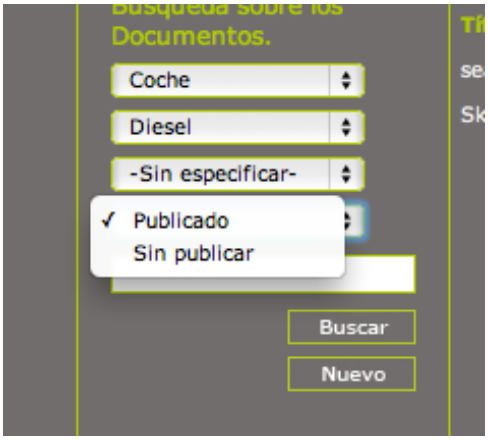
Este buscador nos permite filtrar nuestra búsqueda por diferentes criterios el primero de los cuales es marcar a qué entidad pertenece el documento que estamos buscando.

Dentro del segundo combo será donde podremos filtrar por el grupo al que pertenece nuestro documento siempre que hayamos elegido una entidad en el combo anterior.



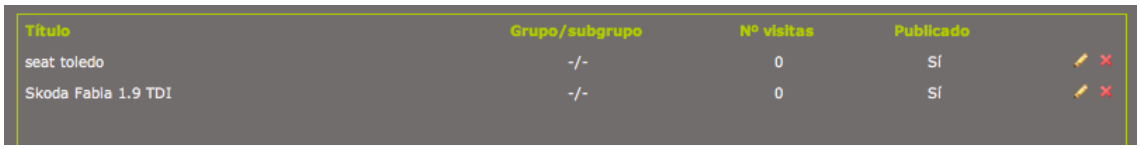
El tercer combo de este buscador corresponde al subgrupo al que corresponde nuestro documento en caso de que tenga uno asignado, es necesario haber seleccionado una entidad del primer combo y un grupo del segundo para poder seleccionar un subgrupo en el tercero.

En el cuarto combo es donde podemos decirle al buscador si el documento que estamos buscando es público o no, este criterio lo único que quiere decir es si el documento en concreto es visible desde la web o no, ejemplo: tengo un documento que es una promoción de verano y sólo es público en esta estación cuando acaba el verano lo despublico y ya no es visible desde la web.



Por último hay un campo de texto en el que podremos escribir cualquier palabra que sepamos que aparece en el documento que estamos buscando. Para realizar la búsqueda una vez finalizado la configuración del filtro es suficiente con hacer clic en el botón Buscar.

El resultado de esta búsqueda se muestra a la derecha del filtro del buscador donde veremos algo de información de cada uno de los documentos que cumplan las restricciones.



El resumen de la búsqueda tiene 4 columnas más el botón de editar y el de eliminar contenido, inicialmente encontramos el título del documento en segundo lugar el

grupo y subgrupo al que pertenece cada uno de ellos. En la tercera columna tenemos el número de peticiones que se ha hecho de ese documento que se refleja como el número de visitas y en la última columna muestra si es público o no.

Independientemente de usar el buscador en la parte izquierda de la ventana tenemos un árbol de contenido agrupado por categorías, desde donde podremos organizar nuestro contenido en carpetas formando una estructura.

En la parte final de este árbol vemos la carpeta agrupaciones, en ésta será donde podamos crear grupos y subgrupos para poder ordenar nuestro contenido y asignar a cada documento un grupo o si es necesario un subgrupo.

Dentro de la sección de agrupaciones también nos encontramos con un simple buscador para poder buscar el grupo que queremos gestionar y un greed a modo de resumen que nos muestra los grupos que se han encontrado.

Este greed de resumen de grupos nos muestra inicialmente la id que tienen en la tabla de la base de datos (esta información es irrelevante para el usuario final pero si es de mucha utilidad para el desarrollador puesto que gracias a esta id podrá recuperar todos los documentos que estén asignados a ésta) En segundo lugar nos

encontramos la descripción o nombre del grupo. En tercer lugar tenemos el campo orden, éste es necesario para que el desarrollador del entorno web pueda recuperar todos los grupos y mostrarlos con el orden que el administrador desee, finalmente tenemos el botón de editar y eliminar el grupo en cuestión.

7.2.1 Mantenimiento de documentos

7.2.1.1 Crear documentos

Para crear documentos tenemos que ubicarnos dentro de la sección contenido, una vez aquí tenemos que seleccionar de nuestro árbol de documentos la entidad de la que vamos a crear un nuevo documento y hacer clic en el botón nuevo.

En esta nueva ventana encontraremos en la parte superior derecha la opción de que este nuevo documente quede asignado a un grupo y al subgrupo que deseemos, un poco más abajo nos encontramos un conjunto de pestañas.

- Castellano: En esta pestaña será donde encontraremos el formulario que tenemos que rellenar con información relevante al documento que estamos creando en este caso (título, marca, modelo, motor, ...). La intención de que esta pestaña se llame Castellano es que en un futuro o en las posteriores versiones (si las hay) de este CMS pueda añadirse la opción de que sea multiidioma con la finalidad de que haya tantas pestañas como idiomas queramos.
- Seguridad: Esta sección de momento deshabilitada servirá en un futuro (en futuras versiones del CMS) para poder poner seguridad a cada documento

por usuario, es decir que al intentar acceder a este documento el propio CMS te saque un formulario de login para acceder a el.

- Documentación relacionada: en esta pestaña será donde relacionaremos un documento con otros documentos independientemente de su temàtica, esta relación es n <-> n, ejemplo: muchas webs que venden aparatos tecnológicos , cuando visitas uno de ellos te dicen qué otros productos han comprado los compradores del producto que estás visitando.
- Documentos dependientes: en esta pestaña es donde he pretendido montar una jerarquía de padres y hijos entre documentos. Un documento puede tener N hijos y 1 padre de esta manera es más sencillo agrupar su contenido en la web independientemente de su temàtica.
- Anexo: para finalizar la sección del anexo es donde podremos subir contenido (imagenes, pdf, etc...) de cada uno de nuestros documentos hasta un máximo de 6. Seria conveniente que en futuras versiones pudiésemos configurar la cantidad de ficheros adjuntos podamos subir a cada documento.

En la parte inferior nos encontramos con un conjunto de botones. El primero de los cuales sirve para imprimir una copia del documento que estamos tratando, el segundo lo utilizaremos para publicar o despublicar un documento, esta opción nunca borrará contenido sólo marcará el documento con una flag para saber si tiene que ser visible desde la web o no. El tercero de los cuales lo utilicaremos para guardar el documento y el último para cancelar el mismo.

7.2.1.2 Modificar documentos

Para modificar un documento previamente creado y guardado será suficiente haciendo clic en la imagen del lápiz o en la misma línea del greed de resumen de documentos.

7.2.1.3 Eliminar documentos

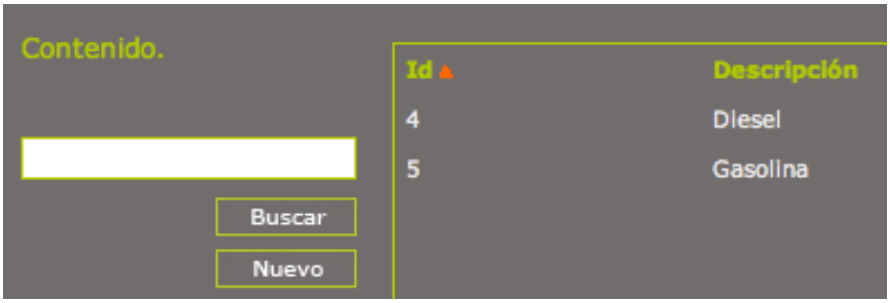
Para eliminar un documento será suficiente hacer clic en la "x" de la línea del documento en cuestión, es importante saber que esta opción no tiene nada que ver con la posibilidad de publicar o despublicar un documento puesto que éstas en ningún momento borrarán información del mismo.

7.2.2 Mantenimiento de agrupaciones

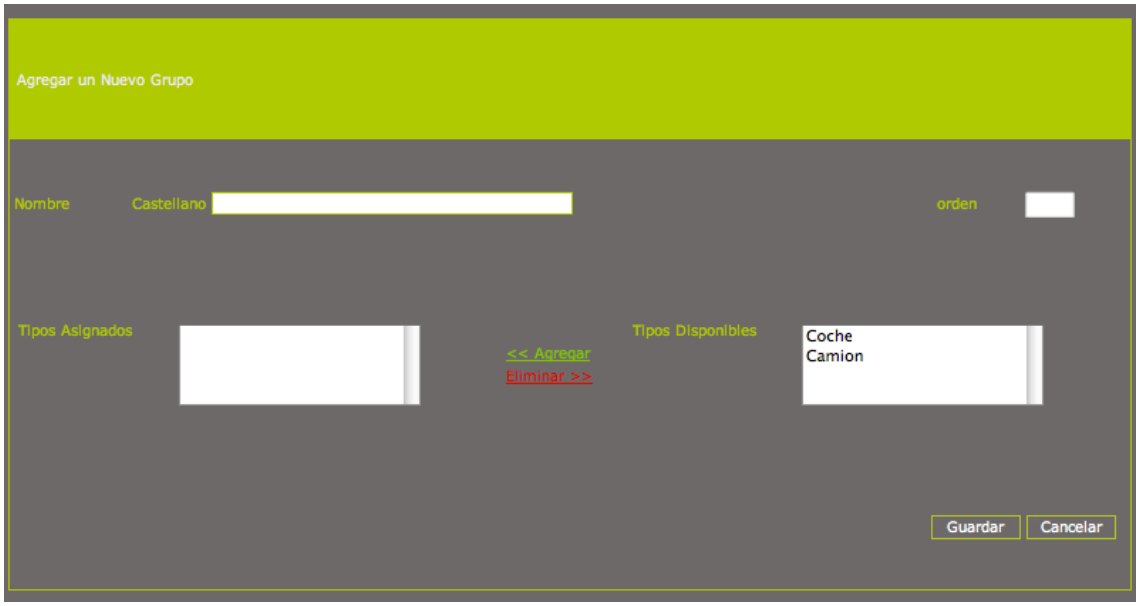
7.2.2.1 Crear agrupaciones

Para crear documentos es necesario que nos ubiquemos en Contenido > Agrupaciones.

Una vez aquí es necesario hacer clic en el botón nuevo para crear un nuevo grupo.



Una vez realizada esta acción nos aparecerá un nuevo formulario para poder configurar diferentes parámetros del grupo que vamos a crear.



En primer lugar debemos rellenar el campo nombre y el orden del grupo (es posible que en la web tengamos que mostrar los grupos y el orden que tengan nos ayudará a mostrar esta información ordenada correctamente), un poco más abajo tenemos que elegir las entidades que podrán optar a este nuevo grupo que estamos creando, ejemplo: el grupo diesel puede afectar tanto a coche como a la entidad camión o puede darse el caso que sólo un entidad disponga de este grupo.

Para añadir o quitar entidades a las que afectará este grupo será suficiente marcándolas en el cuadro de la derecha y haciendo clic en "<< Agregar" lo mismo sucede cuando queremos quitar una entidad ya agregada con el botón de "Eliminar >>"

7.2.2.2 Editar grupos

Para modificar un grupo o agrupación previamente creada y guardada será suficiente haciendo clic en la imagen del lápiz o en la misma líena del greed de resumen de agrupaciones.

7.2.2.3 Eliminar grupos

Para eliminar un grupo será suficiente hacer clic en la "x" de la línea de la agrupación en cuestión.

7.2.3 Mantenimiento de subgrupos

7.2.3.1 Crear subgrupos

Una vez hayamos creado un grupo podremos crear subgrupos de este mismo si los necesitamos, para ellos nos tenemos que dirigir a Contenido >> Agrupaciones es aquí donde debemos seleccionar el grupo donde querremos añadir nuevas categorías para poder organizar mejor nuestra información.

En esta ventana podremos ver con disponemos de una sección inferior donde podremos gestionar los subgrupos. Para crear un subgrupo tenemos que darle clic al botón nuevo, una vez hecha esta acción nos aparecerá un pequeño formulario.

Este formulario dispone de dos campos el primero será el nombre del subgrupo y el segundo el orden para poder gestionar todos los subgrupos de los que dispongamos en un futuro. Para guardar este subgrupo será suficiente con hacer clic en el botón guardar del interior del frame del subgrupo que hemos creado ya que el botón guardar que está situado en la parte superior de la imagen anteriormente mostrada se refiere al grupo y no al subgrupo.

Si todo ha funcionado correctamente tendría que mostrarnos dentro de una caja el subgrupo que acabamos de crear.

7.2.3.2 Editar subgrupos

Para editar un subgrupo debemos hacer clic en el subgrupo deseado y justo a la derecha de la caja que muestra todos los subgrupos se nos abrirá el mismo formulario de creación. Será suficiente con cambiar la información que deseemos y hacer clic en el botón guardar.

7.2.3.3 Eliminar subgrupos

Para eliminar un subgrupo ya creado tenemos que dirigirnos a la caja donde se muestran todos los subgrupos de pertenecientes a un mismo grupo, seleccionar el que deseamos eliminar y hacer clic en el botón borrar.

7.3 Administración

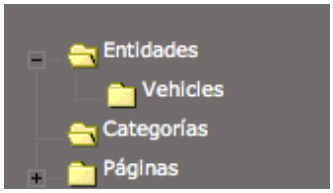
En la sección administración es donde podemos gestionar todos los aspectos de visionado de la información, la gestión tanto de las entidades como de las categorías.

Cuando entramos en la sección de administración podemos ver que el árbol que nos aparece es el siguiente separando claramente cada uno de los conceptos que contiene esta sección.

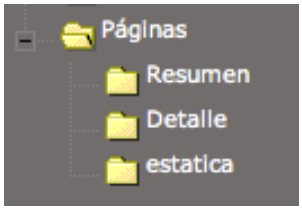
- Entidades: Podemos entender que una entidad es algo parecido a lo que seria una clase en cualquier lenguaje orientado a objetos, así que en la sección de entidades podremos crear entidades con los campos que necesitemos para mostrar inmediatamente después poder crear documentos (que serían siguiendo con el símil del lenguaje orientado a objetos, instancias de la clase concreta) de la entidad que hemos creada. Ejemplo: podemos crear la entidad coche con los campos (marca, modelo, cilindrada,

puertas, etc…) de esta manera podremos crear documentos de esta entidad para poder crear coches.

- Categorías : Una categoría la utilizaremos para agrupar entidades que tengan una misma temàtica por ejemplo en el caso que estamos tratando hemos creado la categoría vehículo en la que hemos agrupado tanto la entidad camión como la entidad coche.



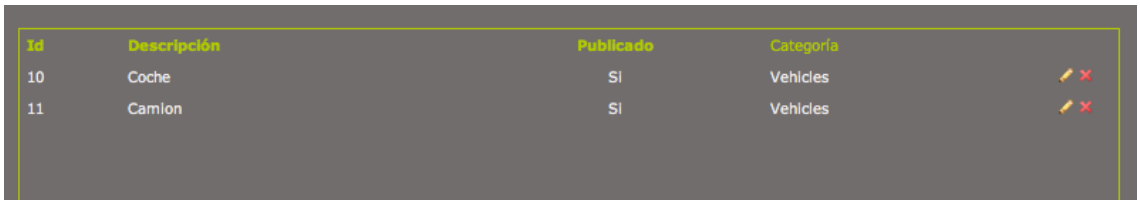
- Paginas: En esta sección es donde podremos crear las diferentes plantillas en HTML que queramos usar para mostrar el contenido del que dispongamos, hay tres tipos de plantillas que podemos crear plantillas de detalle (sólo se trata un documento de una entidad concreta), plantillas de resumen (se tratan los mismos documentos de una misma entidad), plantillas estáticas (serán todas aquellas que su contenido no será dinámico y no se podrá gestionar desde el CMS, cada vez que queramos cambiar algo de estas plantillas tendremos que editar código HTML)



Cada una de estas tres secciones: páginas, categorías y entidades dispone de un buscador inicial que podremos usar para poder buscar la información que necesitemos.

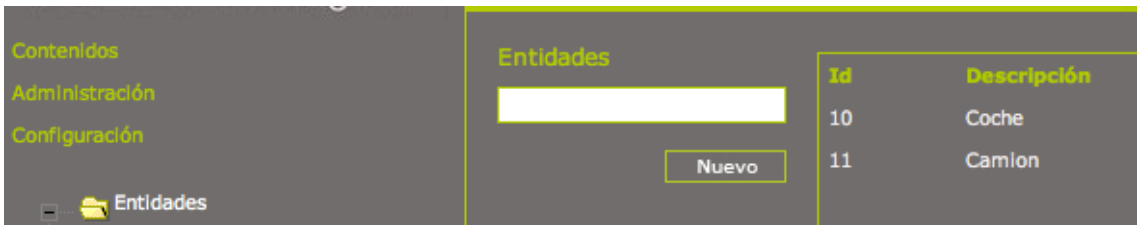
7.3.1 Entidades

Dentro de la sección Entidades inicialmente encontraremos un greed de resumen de todas las entidades que hayamos creado.



7.3.1.1 Crear entidades

Para crear una entidad nueva es necesario hacer clic en el botón nuevo que sale justo al lado del greed de resumen.

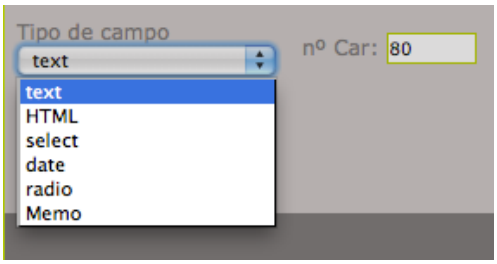


Una vez realizado este paso se nos abrirá en el frame de la derecha un conjunto de pestañas para configurar y generar la entidad.

En esta nueva ventana encontramos dos pestañas, la primera de las cuales es general en la que podremos ponerle un nombre a la entidad que será con la que nos aparezca en el greed de resumen anteriormente comentado. Luego hay un campo orden para poder ordenar las entidades, un campo de path de imágenes que lo utilizaremos para generar una carpeta en el servidor con el nombre indicado con la finalidad de poder subir ficheros adjuntos, en la segunda fila encontramos inicialmente un combo con las categorías que hayamos creado para poder agrupar entidades. Luego vemos dos campos mas actualmente sin uso que son pública y visible, estas dos funcionalidades no forman parte de los objetivos principales del pfc pero tengo intención de que al finalizar su implementación la primera de las cuales sería para añadir seguridad por entidad y la segunda para que no sea visible en búsquedas.

En la pestaña estructura será donde podremos agregar los campos que necesitemos.

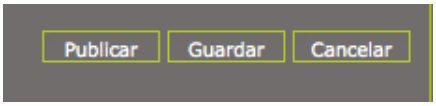
Inicialmente nos creará un campo que será el título con el que nos aparecerá el documento en el greed de resumen de documentos, en esta ventana podremos configurar el nombre del campo, la posición para poder acceder a el desde las plantillas, el tipo de campo que queremos insertar en este caso hemos elegido un campo text de 80 caracteres.



Para agregar más campos será suficiente con hacer clic en el botón agregar y nos aparecerá otra línea como la que está en verde en la foto superior con el nombre de campo 0.

Para poder configurar cada campo tenemos que hacer clic en el botón que contiene un "+" y nos desplegará la ventana para configurar el campo que hemos visto anteriormente, para cerrar el desplegable es suficiente con darle al botón marcado con una "x".

Para guardar la entidad tenemos que ir a la pestaña General y hacer clic en el botón Guardar o en el Publicar es necesario saber que cuando creamos una entidad y hacemos clic en publicar el sistema guardará y publicará la entidad automáticamente.



7.3.1.1.2 Editar entidades

Para editar una entidad previamente creada tendremos que situarnos en el greed de resumen de entidades y localizar la entidad que deseamos editar, para realizar esta acción tenemos que hacer clic en la línea de la entidad o en el lápiz de edición.

7.3.1.1.3 Eliminar entidades

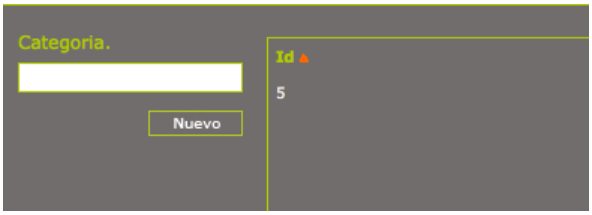
Para eliminar una entidad es suficiente con ubicarnos en el greed de resumen de entidades localizar la entidad que deseamos eliminar y hacer clic en la X roja que esta situada al final del registro.

7.3.2 Categorias

Dentro de la sección de categorías encontraremos un greed de resumen de éstas para localizar la que queremos editar o eliminar en función de las necesidades que tengamos.

7.3.2.1 Crear categorías

Para crear una categoría necesitamos tenemos que hacer clic en el botón nuevo que está situado dentro de la sección de categorías a la izquierda del greed de resumen de las mismas.



Una vez hayamos hecho clic en el botón nuevo nos abrirá dentro del frame de la derecha una ventana para poder configurar esta nueva categoría.

Sólo tendremos que rellenar el campo nombre para poder crear la categoría y hacer clic en el botón guardar.

7.3.2.2 Editar Categorias

Para editar una categoría tenemos que ubicarnos en el greed de resumen de las categorías, localizar la que queremos editar y hacer clic en la línea o en el lápiz de edición.

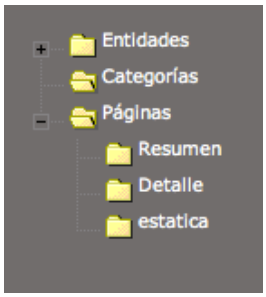
Justo después nos abrirá la misma ventana que abre para la creación y podremos editar la información necesaria.

7.3.2.3 Eliminar Categorias

Para eliminar una categoria es suficiente con darle a la x que hay en rojo al final de la línea del registro de la categoría deseada.

7.3.3 Páginas

Cuando desplegamos la sección Páginas dentro del árbol podemos ver tres subsecciones detalle, resumen y estática.



Cuando hacemos clic en la sección páginas nos aparece un greed de resumen con todas las plantillas que hemos creado.

Id	Descripción	Tipo
3	resumen_servicios	Resumen
4	resumen_promocion	Resumen
5	resumen_noticias	Resumen
6	resumen_proveedores	Resumen
7	resumen_extintores	Resumen
8	resumen_certificados	Resumen
9	detalle_soporte	Resumen
10	resumen_solo_foto	Resumen
11	resumen_buscador	Resumen

7.3.3.1 Paginas resumen

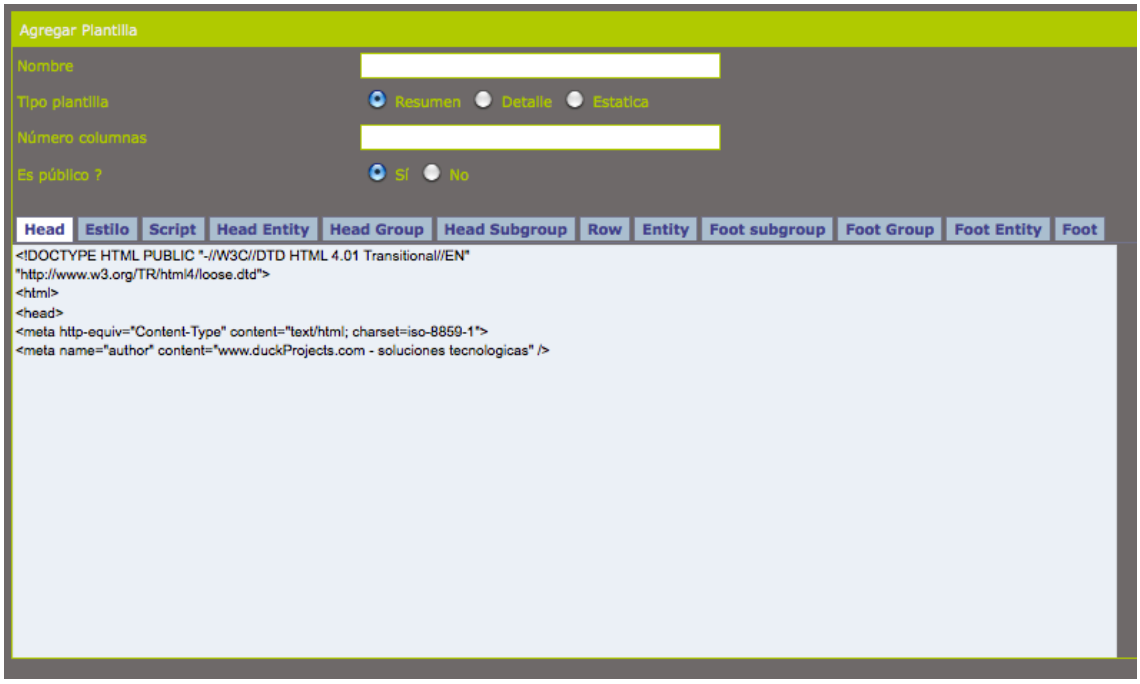
Las páginas de resumen serán las paginas que usaremos para hacer resumen de varios documentos de una misma entidad siguen con el ejemplo que estamos tratando podríamos sacar un resumen de todos los coches que tengamos insertados como documentos.

Para visualizar todas las plantillas de resumen que tengamos es suficiente con hacer clic en el botón resumen y filtraremos en el mismo greed todas las páginas que pertenezcan al grupo de páginas de resumen.

Id	Descripción	Tipo
ructura	resumen_servicios	Resumen
4	resumen_promocion	Resumen
5	resumen_noticias	Resumen
6	resumen_proveedores	Resumen
7	resumen_extintores	Resumen
8	resumen_certificados	Resumen
10	resumen_solo_foto	Resumen
11	resumen_buscador	Resumen

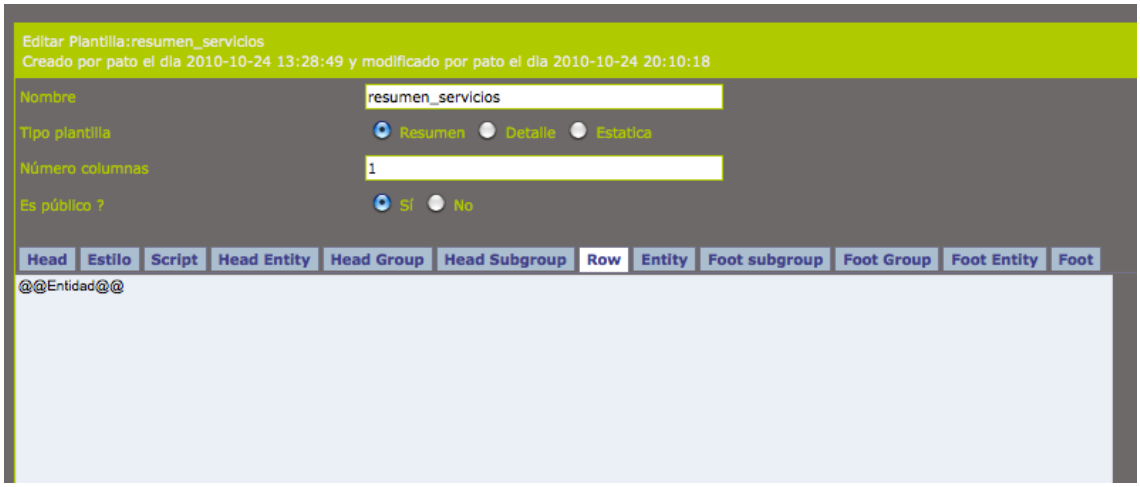
7.3.3.1.1 Crear página de resumen

Para crear una página del tipo resumen el primer paso es hacer clic en el botón nuevo una vez realizada esta acción nos aparecerá una ventana nueva en el frame de la derecha con un conjunto de pestañas que tendremos que rellenar para configurar la página.



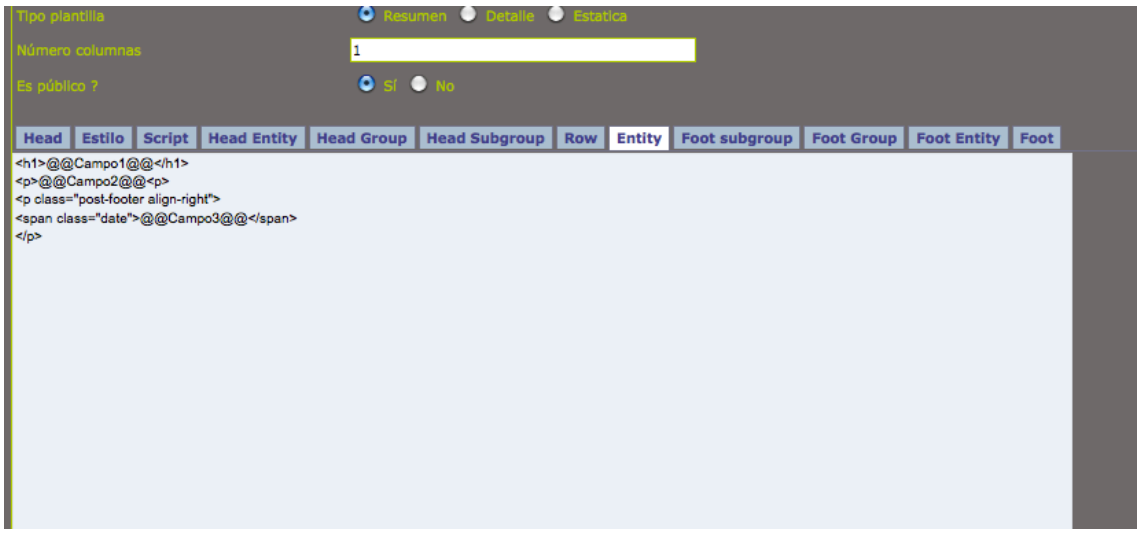
En esta ventana podemos observar que inicialmente tenemos el campo nombre para poder editarla o eliminarla en un futuro, a continuación tenemos unos radio buttons de los cuales tenemos que elegir al que corresponda nuestra página para poder luego mostrarla en la sección que corresponda, en concreto la plantilla resumen tiene el campo numero de columnas para poder realizar diferentes tipos de resumen, con la cantidad de columnas que deseemos. El siguiente campo actualmente no tiene funcionalidad ninguna puesto que hace referencia a si esta página tiene seguridad o es pública. Finalmente en la parte inferior tenemos un conjunto de pestañas las que en conjunto formarán toda la pagina HTML que nos mostrará nuestro navegador. La idea de este conjunto de pestañas es cogida de otro CMS que divide todas sus plantillas en secciones con la finalidad de agilizar su creación y edición.

Entre todas estas pestañas tienen una especial importancia las pestañas "Row" y "Entity" el conjunto y funcionamiento de estas dos estañas es parecido a la de un bucle la pestaña Row dice cuántas veces se tiene que repetir el contenido de la pestaña Entity un ejemplo seria el mostrado en las siguientes imágenes.



En esta imagen por cada columna sólo mostraremos una entidad por eso tenemos que poner dentro de la pestaña Row la cadena @@Entidad@@, en el caso de que tuviésemos que hacer un resumen de más de una columna lo tendríamos que marcar inicialmente en el campo número de columnas y en la pestaña row tendríamos que poner "@@Entidad1@@, @@Entidad2@@, ..." y así sucesivamente hasta cumplir el número de columnas deseadas.

En la siguiente imagen se muestra el contenido de la pestaña entity que será donde se mostrará la información del documento que se esté tratando.



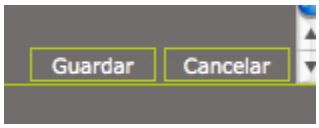
En la imagen anterior podemos observar el contenido de la pestaña Entity y se puede apreciar como se hace referencia a cada uno de los campos de la entidad

(@@Campo1@@, @@Campo2@@, …) así de esta manera podemos hacer referencia a todos los campos que queramos añadiendo al final de la palabra Campo el número de la posición dentro de la entidad del campo que queremos mostrar. Para hacer referencia a las imágenes o ficheros adjuntos tenemos que poner @@Image1@@ o el número que corresponda en cada momento.

```
<h3><a href="#">@@Campo1@@</a></h3>
 <div>
<div class="texto" style="padding-bottom:10px">@@Campo2@@</div>
<img id="img_@@ID@@" class="imagenes_fotos" />
<script>set_image("@@Image1@@","img_@@ID@@");</script>
</div>
```

En esta última foto se puede observar como se insertan imágenes en la plantilla Entity, en esta plantilla también se usa el campo @@ID@@ es la id del propio documento de esta manera nos sirve de ayuda para agilizar el código HTML.

Para guardar nuestra página es suficiente con hacer clic en el botón guardar que se encuentra en la parte inferior de la página.

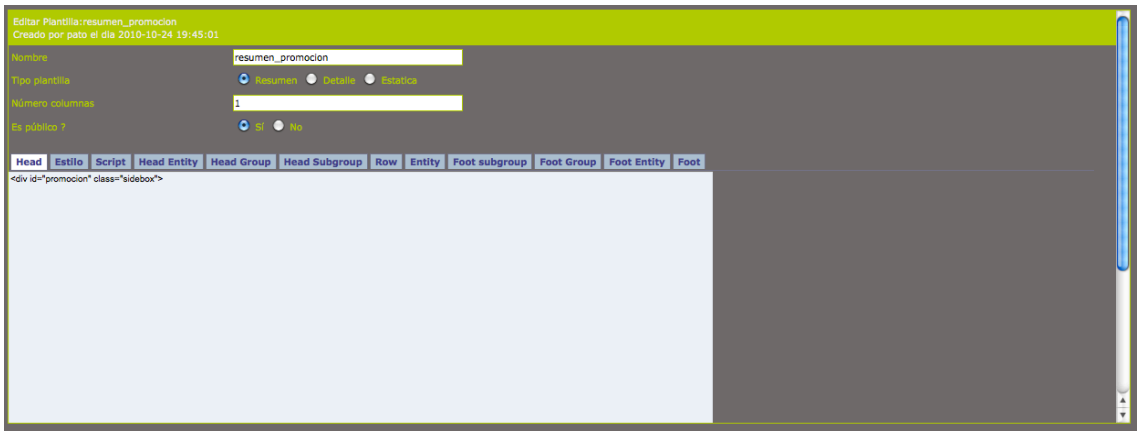


7.3.3.1.2 Editar página de resumen

Para editar una pagina de la sección resumen lo único que tenemos que hacer es localizarla dentro del greed de resumen de éstas y hacer clic encima del registro o clic encima del lápiz de edición.

Id	Descripción	Tipo
3	resumen_servicios	Resumen
4	resumen_promocion	Resumen
5	resumen_noticias	Resumen
6	resumen_proveedores	Resumen
7	resumen_extintores	Resumen
8	resumen_certificados	Resumen
10	resumen_solo_foto	Resumen
11	resumen_buscador	Resumen

Cuando hayamos hecho clic no abrirá la misma página que veríamos si la estuviésemos creado para editar la información que necesitemos.



Para guardar la información que hemos editado es suficiente haciendo clic en el botón guardar

7.3.3.1.3 Eliminar página de resumen

Para eliminar una plantilla resumen será suficiente con hacer clic en la X roja que se encuentra a la derecha del registro que estamos tratando.

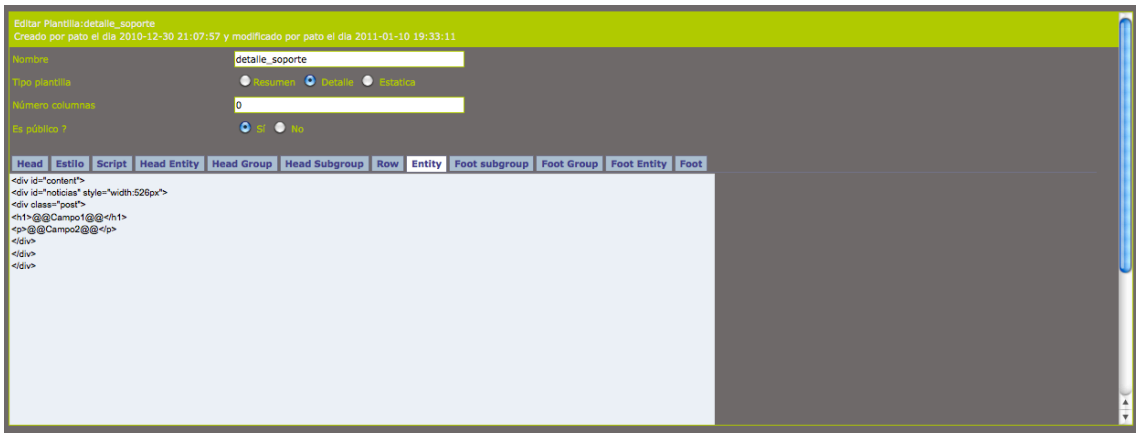
7.3.3.2 Páginas de detalle

Estas páginas las usaremos para ver exclusivamente un documento de una entidad específica siguiendo el ejemplo que estamos tratando en una página de detalle de un coche mostraríamos sólo un vehículo con todos sus campos y imágenes si las tiene.

Cuando hacemos clic en el botón de detalle veremos todas las páginas de las que dispongamos que pertenezcan a esta sección.

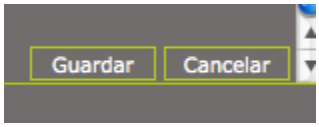
7.3.3.2.1 Crear página de detalle

Para crear una página de detalle tenemos que hacer clic en el botón nuevo y nos aparecerá una página igual a la de la plantilla de resumen, la única diferencia será que en el campo número de columnas tendremos que poner 0 puesto que en una plantilla de detalle sólo trabajamos con un documento y es evidente que no habrá ninguna columna posible.



En estas plantillas de detalle las pestañas row y entity funcionan de la misma manera, con la única diferencia que en la pestaña de Row tendremos siempre sólo una entidad @@Entidad@@ y seguramente el Entity será más extenso que en la plantilla de resumen puesto que en esta será donde expondremos con detalle toda la información de la que dispongamos de este documento.

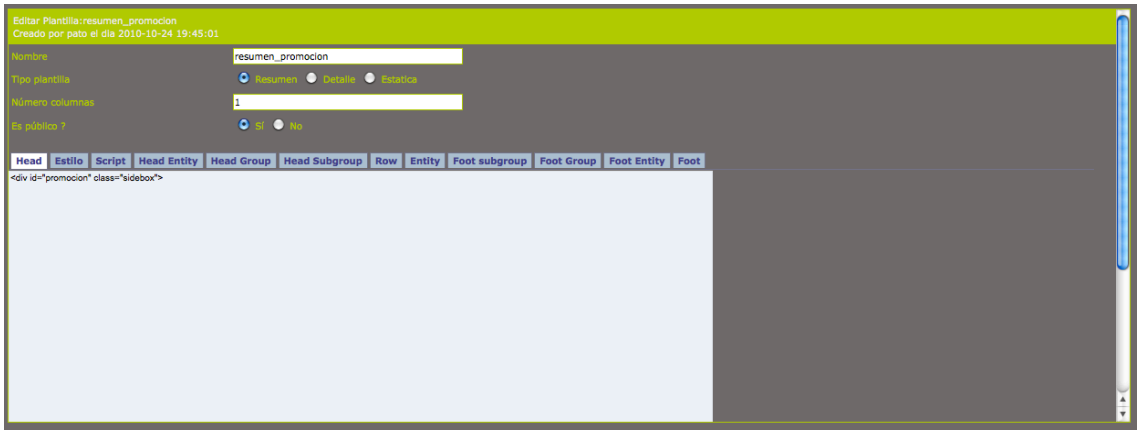
Para guardar nuestra página es suficiente con hacer clic en el botón guardar que se encuentra en la parte inferior de la página.



7.3.3.2.2 Editar página de detalle

Para editar una página de la sección detalle lo único que tenemos que hacer es localizarla dentro del greed de resumen de éstas y hacer clic encima del registro o clic encima del lápiz de edición.

Cuando hayamos hecho clic no abrirá la misma página que veríamos si la estuviésemos creado para editar la información que necesitemos.



Para guardar la información que hemos editado es suficiente haciendo clic en el botón guardar.

7.3.3.2.3 Eliminar página de detalle

Para eliminar una plantilla de detalle será suficiente con hacer clic en la X roja que se encuentra a la derecha del registro que estamos tratando.

7.3.3.3 Páginas estáticas

Las paginas estáticas como su nombre indica serán paginas que la información que reflejan no dependerá de los documentos que enseñen puesto que no trabajan con ningún documento, toda la información que reflejan estas páginas estará insertada directamente en la plantilla que crearemos en el CMS.

Estas plantillas serán las que usaré habitualmente para hacer las partes típicas que tiene cualquier web de una empresa como por ejemplo es la página de contacto que nos muestra información que rara vez cambiará.

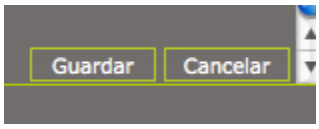
Cuando hagamos clic en el botón estática nos generará un greed con todas las plantillas que pertenezcan a esta sección.

7.3.3.3.1 Crear página estática

Para crear una página estática inicialmente tendremos que hacer clic en el botón nuevo y nos aparecerá una ventana en el frame derecho en la que tendremos que rellenar diversos campos.

Esta ventana es muy parecida a las otras dos con la particularidad de que sólo usaremos la pestaña Head que será donde pondremos todo nuestro código html puesto que al ser estática no tiene sentido utilizar las diferentes pestañas

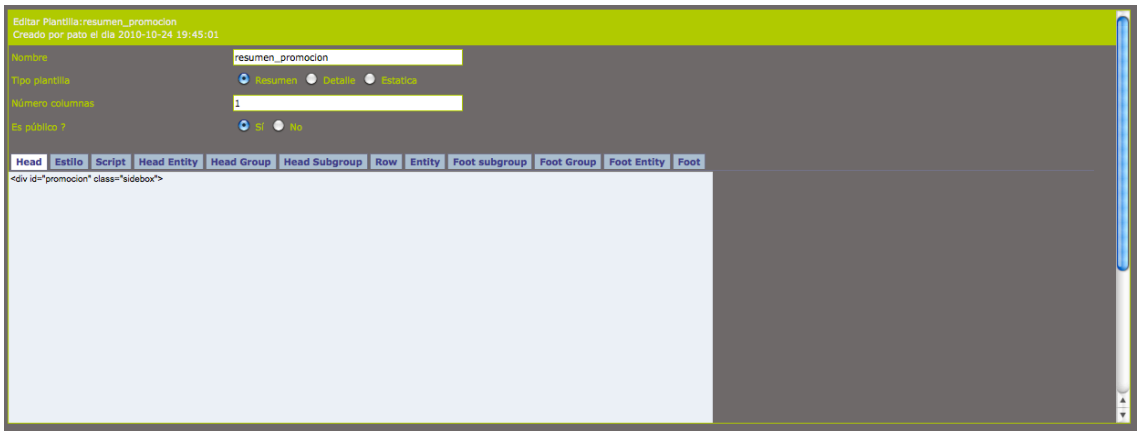
Para guardar nuestra página es suficiente con hacer clic en el botón guardar que se encuentra en la parte inferior de la página.



7.3.3.3.2 Editar página estática

Para editar una pagina de la sección estática lo único que tenemos que hacer es localizarla dentro del greed de resumen de éstas y hacer clic encima del registro o clic encima del lápiz de edición.

Cuando hayamos hecho clic no abrirá la misma pagina que veríamos si la estuviésemos creado para editar la información que necesitemos.



Para guardar la información que hemos editado es suficiente haciendo clic en el botón guardar

7.3.3.3.3 Eliminar página estática

Para eliminar una plantilla estática será suficiente con hacer clic en la X roja que se encuentra a la derecha del registro que estamos tratando.

7.4 Configuración

En esta sección será donde podremos mantener los usuarios que tendrán acceso al CMS, me gustaría que en un futuro, pudiera agregar un sistema de registro web para usuarios web de un portal y gestionarlos por el CMS como usuarios web sin que tengan nada que ver con el propio CMS.

Cuando hacemos clic en usuarios nos aparecerá un greed de resumen de los usuarios de la aplicación.

7.4.1 Usuários

7.4.1.1 Publicador

Este usuario únicamente tendrá permisos para publicar contenido de las entidades creadas, sólo verá la opción de Contenidos. Podrá gestionar los documentos únicamente.

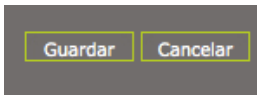
Si hacemos clic en el botón publicador agruparemos todos los usuarios de la aplicación que dispongan de este rol.

7.4.1.1.1 Crear usuario publicador

Para crear un usuario con el rol de publicador tenemos que hacer clic en nuevo y nos aparecerá una ventana en el frame derecho de la pantalla con un conjunto de campos que tendremos que rellenar.

Una vez rellenados los campos relevantes al usuario debemos seleccionar dentro del perfil al grupo que va a pertenecer este usuario.

Para guardar este usuario será suficiente si hacemos clic en el botón guardar que se encuentra en la parte inferior derecha de la página.



7.4.1.1.2 Editar usuario publicador

Para editar un usuario publicador previamente creado tenemos que localizarlo dentro del greed de resumen de los usuarios con este rol una ves localizado debemos hacer clic en el mismo registro o en el lápiz de edición.

Una vez realizada esta acción abriremos una ventana iguala a la de creación de usuarios donde podremos editar la información necesaria y pulsaremos guardar para finalizar la edición

7.4.1.1.3 Eliminar usuario publicador

Para eliminar un usuario es suficiente con localizarlo dentro del greed de resumen y hacer clic en la X de color rojo que esta a la derecha del registro que estamos tratando.

7.4.1.2 Administrador

El usuario administrador no tiene ningún tipo de restricción y puede realizar cualquier operación de las anteriormente comentadas incluso crear otros usuarios de tipo administrador.

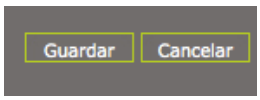
Cuando hacemos clic en el botón administrador en el greed de resumen nos aparecerán todos los usuarios de la aplicación que dispongan de este rol.

7.4.1.2.1 Crear usuario administrador

Para crear un usuario con el rol de administrador tenemos que hacer clic en nuevo y nos aparecerá una ventana en el frame derecho de la pantalla con un conjunto de campos que tendremos que rellenar.

Una vez rellenados los campos relevantes al usuario debemos seleccionar dentro del perfil al grupo que va a pertenecer este usuario.

Para guardar este usuario será suficiente si hacemos clic en el botón guardar que se encuentra en la parte inferior derecha de la página.



7.4.1.2.2 Editar usuario administrador

Para editar un usuario administrador previamente creado tenemos que localizarlo dentro del greed de resumen de los usuarios con este rol una ves localizado debemos hacer clic en el mismo registro o en el lápiz de edición.

Una vez realizada esta acción abriremos una ventana iguala a la de creación de usuarios donde podremos editar la información necesaria y pulsaremos guardar para finalizar la edición.

7.4.1.2.3 Eliminar usuario administrador

Para eliminar un usuario con el rol administrador es suficiente con localizarlo dentro del greed de resumen y hacer clic en la X de color rojo que está a la derecha del registro que estamos tratando.

8 Estructura de la base de datos

```
CREATE TABLE `campos_entidades` (
`idcampos_entidades` int(10) unsigned NOT NULL auto_increment,
`id_entidad` int(10) NOT NULL default '0',
`campo_valor` longtext,
`posicion_campo` int(11) NOT NULL default '0',
`id_campos_tipo` int(10) NOT NULL default '0',
`id_campo_doc_refer` int(11) NOT NULL default '0',
`id_campo_value_refer` int(11) NOT NULL default '0',
PRIMARY KEY  (`idcampos_entidades`,`id_entidad`),
KEY `id_entidad` (`id_entidad`),
KEY `posicion_campo` (`posicion_campo`),
KEY `id_campos_tipo` (`id_campos_tipo`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=181 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `campos_tipo`
--
```

```
CREATE TABLE `campos_tipo` (
`id_campos_tipo` int(10) unsigned NOT NULL auto_increment,
`id_type_field` int(10) unsigned NOT NULL default '0',
`id_type` int(10) NOT NULL default '0',
`campos_tipo_ncar` int(10) unsigned NOT NULL default '0',
`campos_tipo_ncol` int(10) unsigned NOT NULL default '0',
`campos_tipo_value` varchar(255) NOT NULL default '',
`posicion_campo` int(11) NOT NULL default '0',
PRIMARY KEY  (`id_campos_tipo`),
KEY `id_type_field` (`id_type_field`),
KEY `id_type` (`id_type`),
KEY `posicion_campo` (`posicion_campo`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=53 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `categorias`
--

CREATE TABLE `categorias` (
`id_categoria` int(10) unsigned NOT NULL auto_increment,
`cat_name` varchar(255) NOT NULL default '',
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NULL default NULL,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NULL default NULL,
PRIMARY KEY  (`id_categoria`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;
```

```
-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `entidades`
--

CREATE TABLE `entidades` (
`id_entidad` int(10) NOT NULL auto_increment,
`id_type` int(11) NOT NULL default '0',
`ent_publicado` tinyint(1) NOT NULL default '0',
`ftsearch` longtext NOT NULL,
`ent_numfields` int(2) NOT NULL default '0',
`ent_descripcion` varchar(255) default NULL,
`ent_foto1` varchar(100) default NULL,
`ent_foto2` varchar(100) default NULL,
`ent_foto3` varchar(100) default NULL,
`ent_foto4` varchar(100) default NULL,
`ent_foto5` varchar(100) default NULL,
`ent_foto6` varchar(100) default NULL,
`id_grupo` int(10) unsigned default NULL,
`id_subgrupo` int(10) unsigned default NULL,
`ent_titulo` varchar(255) default NULL,
`num_visitas` bigint(20) NOT NULL default '0',
`estado_actual` int(11) default NULL,
`parent_id` int(10) default NULL,
`tiene_hijos` varchar(10) NOT NULL default 'false',
`tiene_relacionados` varchar(10) NOT NULL default 'false',
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NOT NULL default CURRENT_TIMESTAMP,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NOT NULL default '0000-00-00 00:00:00',
```

```
`user_pub` varchar(100) default NULL,
`date_pub` timestamp NOT NULL default '0000-00-00 00:00:00',
`user_last_flow` varchar(100) default NULL,
`date_last_flow` timestamp NULL default '0000-00-00 00:00:00',
`is_public` int(1) default '1',
`id_login` varchar(5) default NULL,
`is_private_group` int(1) NOT NULL default '0',
`tiene_version` char(1) NOT NULL default '0',
`is_version` char(1) NOT NULL default '0',
`ruta` varchar(255) default NULL,
PRIMARY KEY  (`id_entidad`),
KEY `id_type` (`id_type`),
KEY `id_grupo` (`id_grupo`),
KEY `id_subgrupo` (`id_subgrupo`),
KEY `parent_id` (`parent_id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=36 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `entidades_visitas`
--

CREATE TABLE `entidades_visitas` (
`index` bigint(20) NOT NULL auto_increment,
`id_entidad` int(11) NOT NULL default '0',
`timestamp` timestamp NOT NULL default CURRENT_TIMESTAMP on update CURRENT_TIMESTAMP,
`IP` varchar(100) NOT NULL default '',
PRIMARY KEY  (`index`),
KEY `id_entidad` (`id_entidad`)
```

```
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=1961 ;


-- --------------------------------------------------------


--
-- Estructura de tabla para la tabla `formatos_fechas_tipo`
--


CREATE TABLE `formatos_fechas_tipo` (
`id` int(10) NOT NULL auto_increment,
`id_formato_fecha` int(10) NOT NULL default '0',
`id_campo` int(10) NOT NULL default '0',
`id_type` int(10) NOT NULL default '0',
PRIMARY KEY  (`id`),
KEY `id_formato_fecha` (`id_formato_fecha`),
KEY `id_campo` (`id_campo`),
KEY `id_type` (`id_type`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_spanish_ci
AUTO_INCREMENT=168 ;


-- --------------------------------------------------------


--
-- Estructura de tabla para la tabla `formato_fechas`
--


CREATE TABLE `formato_fechas` (
`id_formato_fecha` int(10) NOT NULL auto_increment,
`desc_formato_fecha` varchar(100) collate latin1_spanish_ci NOT NULL default '',
`formato_default` int(11) NOT NULL default '0',
`formato` varchar(10) collate latin1_spanish_ci NOT NULL default 'd-m-yy',
```

```
PRIMARY KEY  (`id_formato_fecha`)
) ENGINE=MyISAM  DEFAULT  CHARSET=latin1  COLLATE=latin1_spanish_ci
AUTO_INCREMENT=4 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `grupos`
--

CREATE TABLE `grupos` (
`id_grupo` int(11) NOT NULL auto_increment,
`grup_orden` tinyint(2) NOT NULL default '0',
`grup_name` longtext NOT NULL,
`ftsearch` longtext,
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NULL default NULL,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NULL default NULL,
PRIMARY KEY  (`id_grupo`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `plantillas`
--

CREATE TABLE `plantillas` (
`id_templates` int(10) NOT NULL auto_increment,
`tmp_popup` tinyint(1) NOT NULL default '0',
```

```
`tmp_name` varchar(255) NOT NULL default '',
`tmp_header` longtext,
`tmp_header_ent` longtext,
`tmp_header_group` longtext,
`tmp_header_subgroup` longtext,
`tmp_row` longtext,
`tmp_entiti` longtext,
`tmp_footer_group` longtext,
`tmp_footer_subgroup` longtext,
`tmp_footer_ent` longtext,
`tmp_footer` longtext,
`tmp_script` longtext,
`tmp_numcol` int(11) default NULL,
`tmp_style` longtext,
`ftsearch` longtext,
`tmp_is_public` int(1) NOT NULL default '1',
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NULL default NULL,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NULL default NULL,
PRIMARY KEY  (`id_templates`),
KEY `templates_name_index` (`tmp_name`),
KEY `id_templates` (`id_templates`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=12 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `procesos`
--
```

```
CREATE TABLE `procesos` (
`id` int(11) NOT NULL auto_increment,
`nombre` varchar(255) NOT NULL default '',
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NOT NULL default CURRENT_TIMESTAMP on update CURRENT_TIMESTAMP,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NOT NULL default '0000-00-00 00:00:00',
PRIMARY KEY (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `registros`
--

CREATE TABLE `registros` (
`id` int(10) unsigned NOT NULL auto_increment,
`nombre` varchar(255) NOT NULL default '',
`apellido1` varchar(255) NOT NULL default '',
`apellido2` varchar(255) default NULL,
`direccion` varchar(255) default NULL,
`cp` varchar(10) default NULL,
`poblacion` varchar(255) default NULL,
`provincia` varchar(255) default NULL,
`pais` varchar(255) default NULL,
`tel1` varchar(10) default NULL,
`tel2` varchar(10) default NULL,
`tel3` varchar(10) default NULL,
`email1` varchar(255) NOT NULL default '',
```

```
`email2` varchar(255) default NULL,
`url` varchar(100) default NULL,
`nick` varchar(20) NOT NULL default '',
`password` varchar(20) NOT NULL default '',
`foto` varchar(255) default NULL,
`flag_mail` smallint(6) NOT NULL default '0',
`id_grupo` smallint(6) NOT NULL default '0',
`id_subgrupo` smallint(6) NOT NULL default '0',
PRIMARY KEY  (`id`),
KEY `id_grupo` (`id_grupo`),
KEY `id_subgrupo` (`id_subgrupo`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `relacion_entidades`
--

CREATE TABLE `relacion_entidades` (
`id_entrada` bigint(20) NOT NULL auto_increment,
`id_padre` int(11) NOT NULL default '0',
`id_hijo` int(11) NOT NULL default '0',
PRIMARY KEY  (`id_entrada`),
KEY `id_padre` (`id_padre`),
KEY `id_hijo` (`id_hijo`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=111 ;

-- --------------------------------------------------------

--
```

```
-- Estructura de tabla para la tabla `rel_tipo_grupo`
--

CREATE TABLE `rel_tipo_grupo` (
`id_type` int(11) NOT NULL default '0',
`id_grupo` int(11) NOT NULL default '0',
PRIMARY KEY  (`id_type`,`id_grupo`),
KEY `id_type` (`id_type`),
KEY `id_grupo` (`id_grupo`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `subgrupos`
--

CREATE TABLE `subgrupos` (
`id_subgrupo` int(11) NOT NULL auto_increment,
`id_grupo` int(11) NOT NULL default '0',
`subg_name` longtext NOT NULL,
`subg_orden` tinyint(2) NOT NULL default '0',
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NULL default NULL,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NULL default NULL,
PRIMARY KEY  (`id_subgrupo`,`id_grupo`),
KEY `id_subgrupo` (`id_subgrupo`),
KEY `id_grupo` (`id_grupo`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=4 ;
```

```
-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `sub_campos_tipo`
--

CREATE TABLE `sub_campos_tipo` (
`id_sub_campos_tipo` int(10) unsigned NOT NULL auto_increment,
`id_campos_tipo` int(10) unsigned NOT NULL default '0',
`subcampo_tipo_value` varchar(155) NOT NULL default '',
PRIMARY KEY  (`id_sub_campos_tipo`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=12 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `tipos`
--

CREATE TABLE `tipos` (
`id_type` int(11) NOT NULL auto_increment,
`id_templates` int(10) default NULL,
`id_templates_popup` int(10) default NULL,
`id_templates_estatica` int(10) default NULL,
`id_categoria` int(10) unsigned default NULL,
`type_menu_style` int(2) default NULL,
`type_submenu_style` int(2) default NULL,
`type_nomfile` varchar(255) default NULL,
`type_prefijo_popup` varchar(255) default NULL,
`type_numcolstemplate` int(11) NOT NULL default '0',
`type_name` varchar(255) NOT NULL default '',
```

```
`type_path` varchar(255) default NULL,
`type_path_popup` varchar(255) default NULL,
`type_path_photo` varchar(100) default NULL,
`type_publicado` tinyint(1) NOT NULL default '0',
`type_numfields` int(2) NOT NULL default '0',
`type_buscable` tinyint(1) NOT NULL default '1',
`type_orden` int(2) NOT NULL default '0',
`ftsearch` longtext,
`type_is_public` int(1) NOT NULL default '1',
`type_id_estado_init` int(11) default NULL,
`type_formula_title` varchar(255) default NULL,
`type_id_login` varchar(5) NOT NULL default '',
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NULL default NULL,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NULL default NULL,
`user_pub` varchar(100) default NULL,
`date_pub` timestamp NULL default NULL,
PRIMARY KEY  (`id_type`),
KEY `types_index_name` (`type_name`),
KEY `id_templates` (`id_templates`),
KEY `id_templates_estatica` (`id_templates_estatica`),
KEY `id_categoria` (`id_categoria`),
KEY `id_templates_popup` (`id_templates_popup`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=12 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `tipo_usuario`
--
```

```
CREATE TABLE `tipo_usuario` (
`id_tipo_usuario` int(2) unsigned NOT NULL auto_increment,
`nom_tipo_usuario` varchar(25) NOT NULL default '',
`privilegios` int(1) NOT NULL default '0',
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NULL default NULL,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NULL default NULL,
PRIMARY KEY  (`id_tipo_usuario`),
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `type_field`
--

CREATE TABLE `type_field` (
`id_type_field` int(10) unsigned NOT NULL auto_increment,
`type_field_name` varchar(45) NOT NULL default '',
PRIMARY KEY  (`id_type_field`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=8 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `usuarios`
--

CREATE TABLE `usuarios` (
```

```
`user_id` int(10) unsigned NOT NULL auto_increment,
`id_tipo_usuario` int(2) unsigned NOT NULL default '0',
`user_email_address` varchar(120) NOT NULL default '',
`user_fname` varchar(120) NOT NULL default '',
`user_lname` varchar(120) NOT NULL default '',
`user_password` varchar(32) NOT NULL default '',
`user_role` varchar(32) NOT NULL default '',
`user_cargo` varchar(100) default NULL,
`user_access` varchar(100) NOT NULL default '',
`user_content_area` varchar(120) default NULL,
`ftsearch` longtext,
`user_crea` varchar(100) NOT NULL default '',
`date_crea` timestamp NULL default NULL,
`user_mod` varchar(100) default NULL,
`date_mod` timestamp NULL default NULL,
PRIMARY KEY  (`user_id`),
KEY `user_email_address` (`user_email_address`),
KEY `user_fname` (`user_fname`,`user_lname`,`user_password`),
KEY `id_tipo_usuario` (`id_tipo_usuario`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=69 ;
```